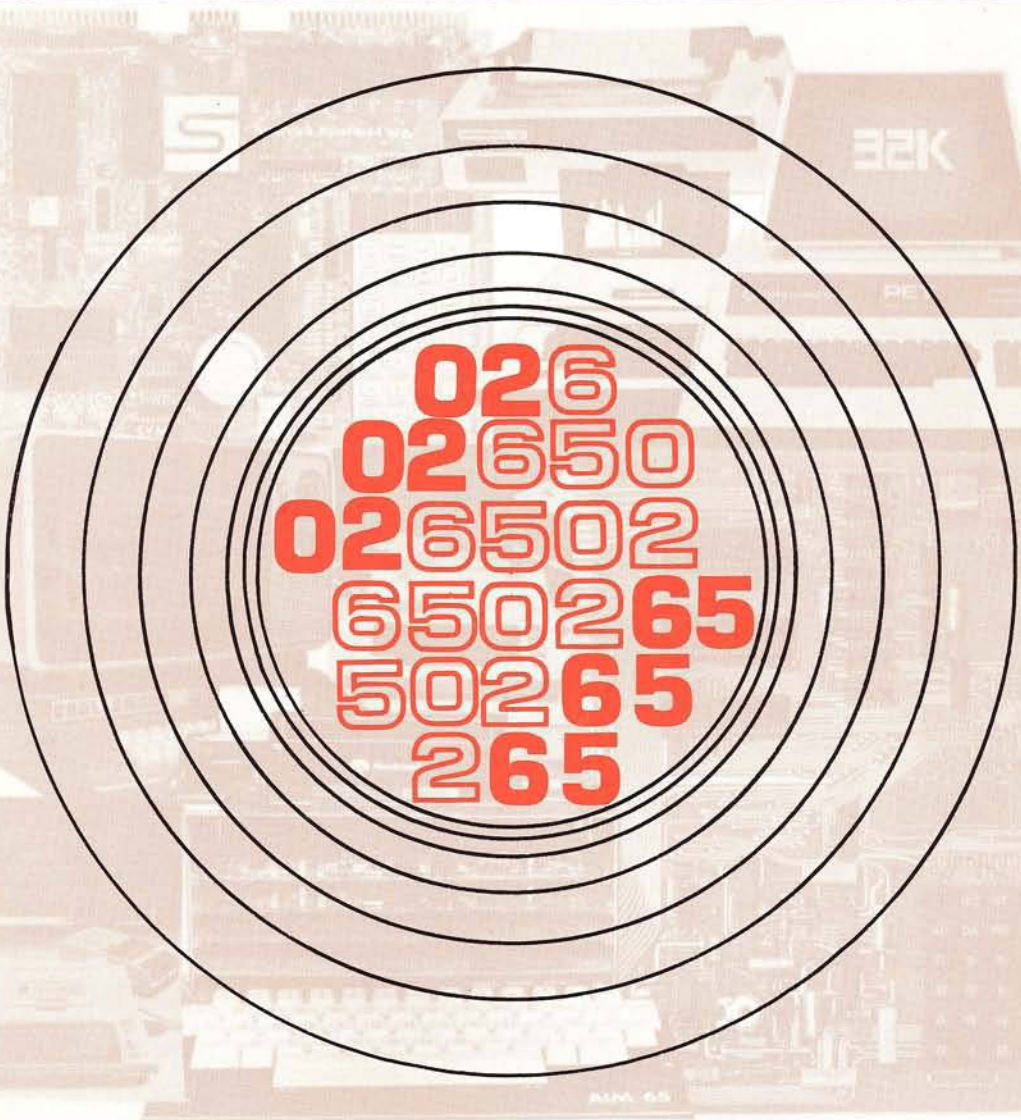
# MICRO

# THE 6502 JOURNAL




NO 24 MAY 1980 $2.00

# GRAND OPENING...SYSTEMS FORMULATE CORPORATION

## Your Microcomputer Information Center

We welcome you to visit our first consumer commercial sales center in the United States. Located in Palo Alto in the Town and Country Shopping Center, Systems Formulate features a full range of microcomputer systems and peripherals for home or commercial use.

During the past few years since our founding in 1978, we have opened three retail stores in Japan stocking premiere American-made microsystems and innovative Japanese systems.

Our new Palo Alto location will feature some exciting surprises for the American interested in microprocessing.

## Miplot, the intelligent plotter ...Only $1,200

Usable even by those with no plotter experience the Miplot is an ideal graphic output device for the personal and small business computer. It's light, compact, and uses ordinary, hard-tip felt pens. (Interchange and cable available)

Special features include:

* Separate pulse motor drives for x and y axis
* Full range of intelligent functions such as line generator and character generator
* Self-test-mode to verify plotter results
* Printer mode for easy interface testing and data dumping

For research, development and management planning, or use by the personal hobbyist...Miplot is the intelligent plotter worth your attention.

Now on display at Systems Formulate.

## Rugged, continuous-feed printers ...Only $1,995

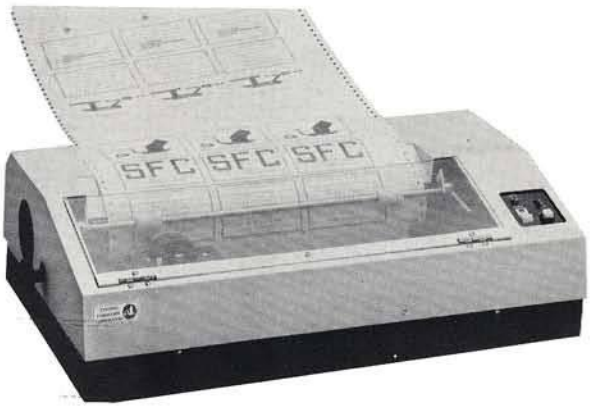
Systems Formulate presents two, new continuous feed printers designed to withstand all the heavy duty use your invoices, statements and statistical analyses can dish out.

Both the SFC-080 and the SFC-136 combine Japanese craftsman ship with many of America's finest components. Like the UMI 8 dot high matrix head, for example.

But that's just the beginning:

* Superior data editing ability
* Your choice of paper size
* Sharp, clear printouts
* Bold double-size characters
* Graphics
* Versitile line spacing
* Paper-saver saving design

Call Systems Formulate and we'll be happy to send you detailed specs or, better yet drop by for a personal demonstration.



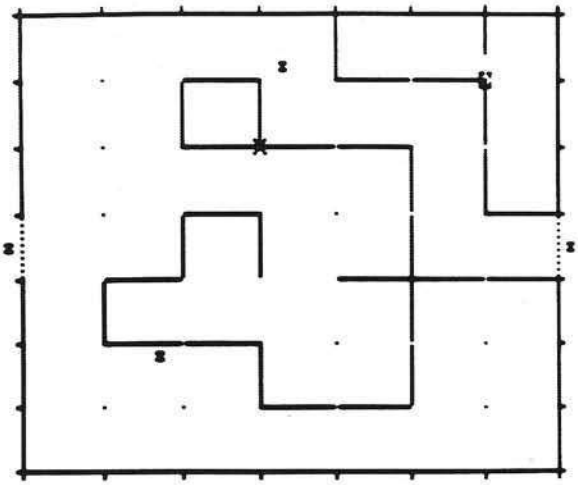

## SYSTEMS FORMULATE CORPORATION

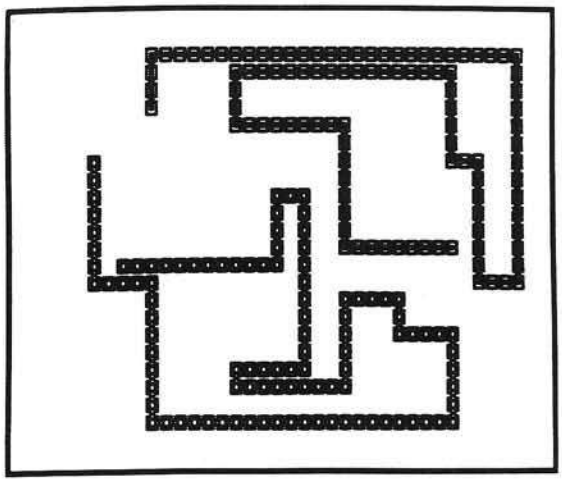
**39 Town & Country Village**
**Palo Alto, California 94301**
**(415) 326-9100**

SYSTEMS FORMULATE CORPORATION

Store hours: Monday-Saturday, 10-7
Sunday, 12-6

# Software for the Apple II




**DYNAMAZE**—a dazzling new real-time game. You move in a rectangular game grid, drawing or erasing walls to reflect balls into your goal (or to deflect them from your opponent's goal). Every ball in your goal is worth 100 points, but you lose a point for each unit of elapsed time and another point for each time unit you are moving. Control the speed with a game paddle: play as fast as ice hockey or as slowly and carefully as chess. Back up and replay any time you want to; it's a reversible game. By Don Stone. Integer Basic (plus machine language); 32 K; $9.95.

**ULTRA BLOCKADE**— the standard against which other versions have to be compared. Enjoy Blockade's superb combination of fast action (don't be the one who crashes) and strategy (the key is accessible open space—maximize yours while minimizing your opponent's). Play against another person or the computer. New high resolution graphics lets you see how you filled in an area—or use reversibility to review a game in slow motion (or at top speed, if that's your style). This is a game that you won't soon get bored with! By Don Stone. Integer Basic (plus machine language); 32 K; $9.95.

**What is a REVERSIBLE GAME? You can stop the play at any point, back up and then do an "instant replay", analyzing your strategy. Or back up and resume the game at an earlier point, trying out a different strategy. Reversibility makes learning a challenging new game more fun. And helps you become a skilled player sooner.**

---

**WORLD OF ODYSSEY**—a new adventure game utilizing the full power of Disk II, which enables the player to explore 353 rooms on 6 different levels full of dragons, dwarfs, orcs, goblins, gold and jewels. Applesoft II 48K; $19.95 includes diskette.

**PERQUACKEY**—an exciting vocabulary game which pits the player against the clock. The object of the game is to form words from a group of 10 letters which the computer chooses at random. The words must be 3 to 10 characters in length with no more than 5 words of any particular length. Each player has only 3 minutes per turn. The larger the words the higher the score. Applesoft II 16K; $9.95.

**APPLESHIP**—is a naval game in which two players enter their ships in respective oceans. Players take turns trying to blast their opponent's ships out of the water. The first player to destroy their opponent's ships may win the game. A great low-res graphics game. Applesoft II 32K; $14.95.

---

**Available at your local computer store**

*Call or write for our free SOFTWARE CATALOG*

Apple II is a registered trademark of Apple Computer, Inc.

**DEALER INQUIRIES INVITED**

***POWERSOFT, INC.***

**P. O. BOX 157**
**PITMAN, NEW JERSEY 08071**
**(609) 589-5500**

*Programs Available on Diskette at $5.00 Additional*

- Check or Money Order
- Include $1.00 for shipping and handling
- C.O.D. ($1.00 add'tl. charge)
- Master Charge and VISA orders accepted
- New Jersey residents add 5% sales tax

HDE inc.

BOX 120
ALLAMUCHY, N.J. 07820
201-362-6574

## HUDSON DIGITAL ELECTRONICS INC.

# ANNOUNCING THE HDE OMNIDISK 65/8

Now, you can "plug in" the latest in a successful series of flexible disk systems developed by HDE for the KIM, SYM and AIM microcomputers. The OMNIDISK 65/8 is a complete system, using 8 inch soft sectored diskettes with a formatted (IBM Standard) capacity of 256K. Of course, a disk formatting function is included as are system supporting utilities for file renaming, disk packing, copy (dual systems) and others.

TED, a full featured, line oriented editor is standard in KIM and SYM based versions to get you up and running on your project in a hurry. The AIM version uses the on-board editor. With the OMNIDISK 65/8 you can concentrate on your problem, the disk supports you all the way.

OMNIDISK 65/8 is available in an attractive walnut wood cabinet, or unpackaged for OEM applications in dual and single drive configurations. The HDE disk controller is a state-of-the-art 4½" by 6½" card electronically compatible with the 44-pin KIM-4 bus structure. The controller and disk-driver are designed to operate with the popular Shugart 801-R and compatible devices.

The OEM single drive is $1195, the dual, $1895 and the dual in the walnut cabinet, $2200. Price is another reason to step up to the proven quality of an HDE system.

**HDE PRODUCTS – BUILT TO BE USED WITH CONFIDENCE**
**AVAILABLE DIRECT OR FROM THESE FINE DEALERS:**

JOHNSON COMPUTER
Box 523
Medina, Ohio 44256
216-725-4560

ARESCO
P.O. Box 43
Audubon, Pa 19407
215-631-9052

PLAINSMAN MICROSYSTEMS
Box 1712
Auburn, Ala. 36830
800-633-8724

LONE STAR ELECTRONICS
Box 488
Manchaca, Texas 78652
612-282-3570

PERRY PERIPHERALS
P.O. Box 924
Miller Place, N.Y. 11764
516-744-6462

MICRO™

MAY 1980
Issue Number 24

## Staff

**Editor/Publisher**
Robert M. Tripp

**Associate Editor**
Mary Ann Curtis

**Circulation Manager**
Carol A. Stark

**Art/Advertising Coordinator**
Terry Spillane

**Comptroller**
Donna M. Tripp

**Production Assistant**
L. Catherine Bland

MICRO™ is published monthly by MICRO INK, Inc., Chelmsford, MA 01824. Tel. 617/256-5515.
Second Class postage paid at Chelmsford, Ma 01824.
Publication Number: COTR 395770.
Circulation: Paid subscriptions: U.S.: 3800, Foreign: 350; Dealers: U.S.: 4500, Foreign: 1900.
Subscription rates: U.S.: $15 per year. Foreign, surface mail: $18 per year.
For air mail rate, change of address, back issue or subscription information write to: MICRO, P.O. Box 6502, Chelmsford, MA 01824.
Entire contents Copyright © 1980 by MICRO INK, Inc.

## Table of Contents

**More About 16 Bits** 5
**Roadrunner: A Math Drill for 2nd Graders** 7
by Peter A. Cook
**Plotting with Special Character Graphics** 11
by Dale DePriest
**SYM-1 BASIC 'GET' Command** 15
by George Wells
**A Simple Temperature Measurement Program & Interface** 19
by Marvin L. DeJong
**Shorthand Commands for Superbaord II and Challenger C1P BASICs** 25
by Henk J. Wevers
**A Formatted Dump Routine for the AIM—65** 29
by W.E. Wilson
**New & Better PET User Port Printer Routines** 33
by Michael Tulloch
**Microbes & Updates** 39
**Graphics and the Challenger C1P, Part 5** 41
by William L. Taylor
**Lower Case and Punctuation in Applesoft** 45
by J.D. Childress
**SYM—1 Sends Morse Code** 49
by Ralph R. Orton
**An EDIT Mask Routine in Applesoft BASIC** 53
by Lee Reynolds
**PET Keysort Update** 59
by Rev. James Strasma
**Expand KIM-1 Versatility in Systems Applications** 63
by Ralph Tenny
**MICRO Club Circuit** 66
**The MICRO Software Catalogue: XX** 69
by Mike Rowe
**6502 Bibliography: Part XX** 72
by William R. Dial
**Index to Vol. 3** 77

## Advertiser's Index

| | | | |
|---|---|---|---|
| Aardvark | 24 | MICRO | 79 |
| AB Computers | 40 | Micro Austin | 37 |
| Avant Garde | 5 | Micro Software | 40 |
| Baclan | 40 | Micro Technology Unlimited | 61 |
| Beta Computer Devices | 65 | Muse | 52 |
| CJM-Industries | 4 | NIBBLE | 10 |
| Classified Ads | 17,27 | On Line | 80 |
| Compute | 68 | Perry Peripherals | 65 |
| Computer Corner of N.J. | 62 | Powersoft, Inc. | 1 |
| The Computerist, Inc. | 31,40 | Programma International | BC |
| Computer Shop | 76 | Progressive Computing | 24 |
| Computer Shopper | 47 | Progressive Computer Software | 76 |
| Connecticut microComputers | 23,60 | Progressive Software | 71 |
| C.R.A.E. | 9 | Rainbow Computing | IBC |
| Cyberdyne | 30 | RNB-Enterprises | 48 |
| Dakin 5 | 13 | Shepardson Microsystems | 6 |
| Decision Systems | 65 | Silver Spurr | 43 |
| Discount Data Products | 24 | SKYLES Electric Works | 14,32 |
| Eastern House Software | 58 | Small Business Computer Sys. | 62 |
| Educational Software Professionals | 47 | Softside Publications | 44 |
| Electronic Specialists, Inc. | 40 | Southeastern Software | 38 |
| Enclosures Group | 28 | Southwestern Data Systems | 9 |
| Excert | 18 | Synergistic Software | 57 |
| Galaxy | 57 | Systems Formula | IFC |
| Holtzman | 76 | TEK Aids | 24 |
| Hudson Digital Electronics | 2 | United Software of America | 37 |

# Now You Can Have
# INPUT/OUTPUT
## For Your Apple Computer

DUAL AXIS JOYSTICK AND PUSHBUTTON MODULE for Games, Graphics, and Experimental Program Input

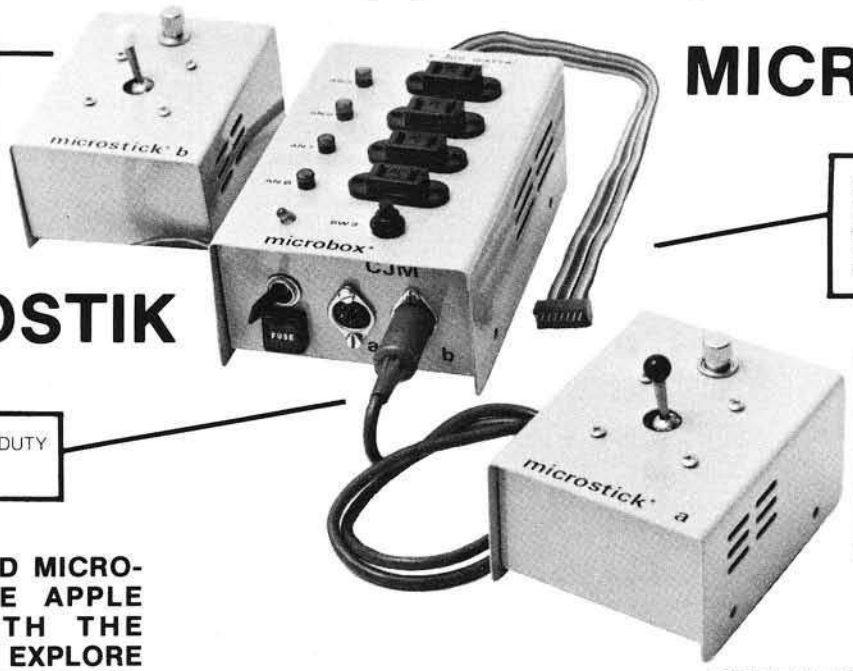
**MICROBOX**

CONNECTS DIRECTLY to the Apple Game Socket. ACCEPTS 2 Dual Axis MICROSTIKS or 4 Paddles

**MICROSTIK**

EXTRA LONG, HEAVY DUTY cables and connectors

ISOLATED SWITCHING of 4 AC loads or relays from a basic program. 4 LED status indicators. Toggle Switch input (sw3)




**MICROBOX AND MICROSTIK PROVIDE APPLE OWNERS WITH THE HARDWARE TO EXPLORE THE INPUT/OUTPUT CAPABILITIES OF THEIR COMPUTERS.**

A SIMPLE COMMAND from the Apple keyboard or a Basic Program can switch an external device. Connect AC loads, such as lamps, motors, relays or solenoids directly through the MICROBOX's 4 AC OUTLETS. Loads can range from 0 to 220VAC and draw up to 200 Watts each. Solid State Switching ISOLATES the load from your Apple for complete safety. Four LEDs provide a visual on/off status of each load.

A Complete Instruction/Tutorial Manual is included with the MICROBOX.

**REAL-TIME INPUT**

The MICROSTIK is a sturdy, two axis joystick. Metal Cable Connectors assure trouble free usage over time, and enable extension cables to be added easily. Use the MICROSTIK to add real-time input to your game, graphic or experimental programs. Each MICROSTIK contains a PUSHBUTTON for added input possibilities.

**MICROBOX and MICROSTIK sit comfortably on, or aside the Apple Computer. They have been designed to match the Apple in color and design.**

**ORDER TODAY AND CONNECT YOUR APPLE TO THE OUTSIDE WORLD.**

The MICROBOX and MICROSTIK can be purchased at most computer stores or can be ordered directly by mail or through our convenient 24 hour telephone service.

TELEPHONE:
(703) 471-4291

Order the MICROSET and receive the MICROBOX, 2 MICROSTIKS, the Manual and Cassette, and SAVE $25.

| | |
|---|---|
| **MICROBOX** | **$109.95** |
| **MICROSTIK (each)** | **34.95** |
| **Demo Cassette** | **9.95** |
| **MICROSET** | **164.95** |
| **12 ft. Ext. Cables** | **6.95** |
| **Relay Modules** | **WRITE** |
| **Solenoid Modules** | **WRITE** |

**Va. residents add 4% sales tax**

**MASTER CHARGE, VISA accepted**
**NO C.O.D.s**

CJM Industries, Dept. MB
316B Victory Drive
Herndon Industr Park
Herndon, Va. 22070

CJM
INDUSTRIES

## More About 16 Bits

Last month's article by Randall Hyde, "The SY6516 Pseudo-16 Bit Processor" [MICRO 23:36] is an interesting combination of a year-old rumor and author fantasy.

More than a year ago, Synertek considered developing a part to be called the SY6516. Several different SY6516's were proposed which significantly differed from each other. It was decided not to develop any of the proposed 6516's. Therefore, it is not "almost ready to ship".

Of course, Synertek does have several other development programs running in both the peripheral and CPU areas. The SY6545 improved CRT controller and the SY6591 floppy disk controller, each with the 6502 bus, will be available later this year.

This letter may not be the action referred to in last month's editorial. A simple phone call to Synertek from Micro to discuss the SY6516 would have forestalled publication of this article which has created so much confusion and annoyance among 6502 enthusiasts.

Michael Smolin
Strategic Marketing Manager,
Synertek, Inc.

*Editor's Note: The intent of the above mentioned article, and my editorial* "The Value of 16 Bits"[MICRO 23:9], *was to spark reader interest in improved versions of our 6502 microprocessor. My intent was* **not** *to cause anyone* "confusion and annoyance" *and to the degree that this has occurred, I apologize. I did, by the way, attempt to get some information from my local distributor, but without any success. Now that I have a contact at Synertek who is aware of this type of project, I will certainly check out any Synertek related material in the future.*

*It is heartening to hear that the article and editorial did generate interest in an improved 6502. Several readers have written with their suggestions. If you have any ideas, please send them in. We will present the best ideas in an article in a few months.*

Robert M. Tripp


# WHAT'S THE ONE THING NO ONE HAS THOUGHT ABOUT DOING WITH COMPUTERS?

We acknowledge that computers are the most valuable data processing devices ever conceived for business and education, and are the most creative toys on earth. However, the potential of computers has only begun to be explored. Avant-Garde Creations has discovered and developed a way to use computers in the areas of self-transformative experiences, life-awareness, making relationships work, and "getting your act together".

Previously, it was thought that such trips as est, Lifespring, Actualizations, and others were the only means of significantly dealing with the above areas. We acknowledge that they are indeed valuable experiences. But because one has to devote many full days and hundreds of dollars to such trips, all those people who aren't yet ready to get into all this that deeply are left with nowhere to turn for such awareness experiences.

The intention of Avant-Garde Creations is to change that. We have the knowledge and techniques, and now we have the programs. The first is called "The Life Dynamic Transformation Experience", and the second is "The Relationship Life Dynamic". Both are available on disk at this time. Over the next year 9 more life dynamic areas programs will be developed in the following areas: physical, creativity, environment, meaning, conditioning, sexuality, normalcy, aliveness, and responsibility.

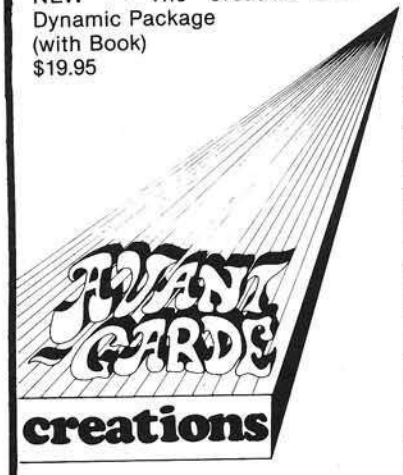
Why spend hundreds of dollars to find out if you're into working on such areas of your life? Why not get something you can use over and over, at your convenience?

One of the most exciting aspects of our programs is that in many program sections, it was discovered that "game-playing" was the most effective method of getting people to perceive a difficult truth. So from time to time you're given 100% original games to play, games specifically designed to give you the opportunity to "get" something challenging or elusive. So even if you don't "get" something, you'll have a great time!

Apple II with Applesoft and 48K required with single disk drive.

*The Life Dynamic Transformation Experience*. . $15.95 postpaid
*The Relationship Life Dynamic*. . . . . $15.95 postpaid
Complete Program Listings available:
*The Life Dynamic Transformation Experience*. . . $4.95 postpaid
*The Relationship Life Dynamic*. . . . . $4.95 postpaid

NEW — The Creative Life Dynamic Package (with Book) $19.95




If you don't have an Apple II with Applesoft, there are approximately 5 Apple stores in every state. We'll send you the address of the one nearest you. Your Apple dealer will be glad to give or rent you (very cheaply) some time on one of their computers, and will be happy to get you started running the program. In case you think you need to understand computers to run our programs, we assure you that if you can read English and type your name, you won't have any problem.

**AVANT-GARDE CREATIONS**
PO Box 30161 Eugene, OR 97403 Dept. C1

Announcing...

OPTIMIZED SYSTEMS SOFTWARE

UPGRADE YOUR APPLE II® WITH A NEW SYSTEMS SOFTWARE PACKAGE

- Unified Operating System
- Disk File Manager
- Commercial Basic
- Editor/Assembler/Debugger
- Data Base Manager

Optimized Systems Software does not use Apple DOS®. OSS is a unified and complete systems software package with its own Operating System and File Manager. The Operating System, the File Manager and the Basic combined use only slightly more RAM than Apple DOS® alone. Requires 48K Apple II® with Disk II.

**Operating System**

- Byte and Block I/O
- Simple User Interface
- Simple Device Interface
  (create your own)

**File Manager**

- Open, Read, Write, Delete, Lock, etc.
- Random Access via Note & Point
- File Names of Primary.Ext type

**Basic**

- Nine Digit Precision DECIMAL Floating Point
- 32K Byte Strings
- Variable Names to 256 significant characters
- I/O Interface Statements
  (no PRINT "control-D...")

**Editor/Assembler/Debugger**

- Line Editor
  (Edits Basic programs, too)
- Mini Assembler
- Maxi Assembler
- Disassembler
- Step, Trace, etc.

**Available NOW**

| | |
|---|---|
| • Operating System + File Manager + Basic | $69.95 |
| • Operating System + File Manager + ASM | $69.95 |
| • Operating System + File Manager + Basic + ASM | $109.95 |
| • Operating System + Data Base Manager | (2nd Q) |

Order today. Add $2.00 for shipping & handling. California residents add 6% sales tax. Visa/Mastercharge welcome. Personal checks require 2 weeks to clear.

Note: Apple II®, Apple DOS® are trademarks of Apple Computer, Inc.

Optimized Systems Software
Shepardson Microsystems, Inc.
20395 Pacifica Dr., Suite 108A
Cupertino, CA 95014
(408) 257-9900

# ROADRUNNER - A Math Drill for Second - Graders

**Remember 'rationalizing' the purchase of your microcomputer on the grounds that 'it would be good for the kids'? Well, he are some suggestions on using your Apple for Computer Assisted Instruction.**

**Peter A. Cook**
**1443 N. 24th Street**
**Mesa, AZ 85203**

Computer Assisted Instruction (CAI) will spread rapidly in the field of education as the use of small computers becomes more widespread. Children in their earliest school years can enjoy using a keyboard to learn the traditional skills, if programs are prepared with imagination and care. The greatest challenge in designing these programs is the fact that young children too easily become bored and lose interest.

### Making a Game of It

My second-grader was having trouble beating the clock during timed math drills at school, so I devised a program which presents him wih 50 practice problems on an Apple II computer. Random additon and subtraction problems such as the following are displayed on the screen.

```
 11     9     3    12
- 2   - 5   + 4  + 0
____  ____  ____  ____
```

The largest number used is 12, for either operand or result. This can be changed as the needs of the child dictate.

To increase the interest level and heighten the student's motivation, several elements have been added which turn the drill into a game, called "Roadrunner". These added features provide the child with a visible goal, immediate reinforcement, and a certain amount of pressure.

Goal orientation is provided by six animal names which illuminate in sequence as each group of ten problems is completed. The lowest level is the snail, for the beginning problems with a long time interval, progressing to the roadrunner for the last problem with the shortest interval.

The player is informed at the end of each problem whether or not his answer is correct. If so, the next problem is presented. If not, the problem sequence is stopped, and the player is given as many chances as needed to determine the right answer. The game will not end until the correct answer has been given, so the student isn't left hanging as to what the proper answer should be.

Pressure is applied in the form of a time countdown for each problem. The time interval starts at 20 seconds for the first ten problems, then decreases by two seconds after each set of ten problems is completed.

The game stops running if an answer is not given before the time elapses, or when an answer is incorrect. Once the correct answer is supplied, or when 50 problems are completed, the animal name corresponding to the highest level attained begins flashing. This is accompanied by several beeps from the speaker, to officially announce the end of the game.

### The Program

Coding the Applesoft program was fairly straightforward with the exception of the time countdown. A random number is first selected between 0 and 24. If it is greater than 12, then 12 is subtracted from it, and the problem will be a subtraction problem. A second random number is then selected between 0 and 12. In the case of an addition problem, the sum of the two numbers is checked to be sure it is no higher than 12. For a subtraction problem, the second number is checked to be sure it is less than the first number. The problem is then presented on the screen, and a reply is requested.

The usual method for inputting a reply to a question in Applesoft, using the INPUT or GET statement, will not work in this case because they cause the program to stop and wait indefinitely until a reply is keyed in. Memory location -16384 contains the ASCII value of the last key depressed, plus 128, provided that the keyboard strobe (-16368) has been reset to zero.(*Applesoft BASIC Programming Reference Manual*,Apple Computer Inc., 1978) By PEEKing this location repeatedly during a time delay loop, as in program lines 60 through 66, the computer will know whether or not an answer has been keyed in prior to the time interval elapsing. Since a reply could have one or two digits, the return key is used to signify the end of data input and to stop the timer from counting down.

Once the reply is received, it is tested for correctness, and the appropriate message is printed. When ten problems have been completed, the animal name and time delay are changed, and the above process is repeated.

Several changes can be made to the program to make the game more or less challenging, depending on the age and ability of the child. Numbers larger or smaller than 12 can be programmed in by changing H in line 2. The time interval, 20 seconds, is defined by T in line 4, and the decrease after each ten problems, 2 seconds, is subtracted from U in line 84.

One caution needs to be mentioned which could cause some frustration if not explained before using the program. The process of reading the keyboard using PEEK (-16384) depends on the particular time during the cycle that a key is depressed, and for how long. It seems to work about 95 per cent of the time. Watch the screen to be sure each digit is printed before pressing the next key during the time countdown, or an incorrect answer will be accepted. Sometimes the desired key must be pressed one or two additonal times. The time interval is purposely long enough to allow for this. Also, if the wrong key is pressed, you cannot back up and correct it.

That completes the description. Type in the program, have your second-grader RUN it, and see if he can answer the problems fast enough to become a ROADRUNNER.

```
]LIST0
0  REM  "ROADRUNNER", PETE COOK,
     OCT 79
2 H = 12: REM HIGHEST NUMBER
4 T = 20: REM LONGEST TIME, SECON
     DS
6  DIM W$(6): FOR W = 1 TO 6: READ
     W$(W): NEXT
8  DATA SNAIL, TURTLE, CHIPMUNK, RAB
     BIT, COYOTE, ROADRUNNER
10  REM  PRINT HEADINGS:
12  HOME : HTAB 11: PRINT "R O A
     D R U N N E R": PRINT : HTAB
     9: PRINT "50 ADD/SUBTRACT PR
     OBLEMS"
14  POKE 34, 5: REM TOP MARGIN
16  POKE 33, 22: REM WIDTH, ALTERN
     ATES FROM LEFT HALF TO RIGHT
      HALF
18 P = 1:X = 0:Y = 1:Z = 0:U = T:
      REM RESETS VARIABLES FOR NE
     W GAME
20  REM  PRINT ANIMALS:
21  VTAB 10: FOR W = 1 TO 6: IF W
      = Y THEN  INVERSE : IF Z =
     1 THEN  FLASH
22  PRINT W$(W): NORMAL : PRINT :
      NEXT
23  IF Z = 1 THEN  FOR C = 1 TO 5
     : FOR D = 1 TO 10: NEXT D: PRINT
      CHR$ (7);: NEXT C: GOTO 90:
      REM  5 BEEPS
24  REM  BLANK LAST PROBLEM, PRIN
     T NEW NUMBER AND TIME REMAIN
     ING:
25  VTAB 6: CALL  - 868: REM  BLA
     NK LINE
26  POKE 32, 17: REM  LEFT MARGIN
27  FOR C = 1 TO 20: FOR D = 1 TO
     60: NEXT D: CALL  - 912: NEXT
     C: REM  SCROLL UP ONE LINE
28  POKE 32, 0: VTAB 6: PRINT "NUM
     BER: ";P: POKE 32, 17: FOR D =
     1 TO 1000: NEXT : REM  DELAY

29  VTAB 6: HTAB 3: PRINT "SECOND
     S: ";U
30  REM  SELECT NUMBERS:
31 S$ = "+ ":L = H:A = 0
32 M =  INT ( RND (1) * 100): IF
     M > 2 * H THEN 32: REM TOP N
     UMBER
34 N =  INT ( RND (1) * 100): IF
     N > H THEN 34: REM BOTTOM NU
     MBER
36  IF M > H THEN 42: REM SUBTRAC
     T
38 S = M + N: IF S > H THEN 34
40  GOTO 46
42 L = M - H: IF N > L THEN 34: REM
     TOP NUMBER MUST BE LARGER
44 S = L - N:S$ = "- ":M = L
45  REM  PRINT PROBLEM:
46  FOR D = 1 TO 1000: NEXT : REM
     DELAY
50  VTAB 9: HTAB 8: IF M < 10 THEN
      HTAB 9: REM RIGHT JUSTIFY
52  PRINT M: HTAB 6: PRINT S$;
54  IF N < 10 THEN  HTAB 9
55  PRINT N: HTAB 6: PRINT "----"
56  REM  INPUT ANSWER, TEST IT:
57 V = U * 18: REM  MULTIPLIER FO
     R ACTUAL SECONDS
58  PRINT : PRINT : HTAB 3: PRINT
      "ANSWER? ";: REM  NO CURSOR
60 I =  PEEK ( - 16384): POKE  -
     16368, 0: REM  READ KEYBOARD,
      RESET KEYBOARD STROBE
62  IF I = 141 THEN  VTAB 15: GOTO
     74: REM  RETURN KEY INDICATE
     S ALL DIGITS RECEIVED
64  IF I > 127 THEN A = A * 10 +
      VAL ( CHR$ (I - 128)): VTAB
     14: HTAB 11: PRINT A: REM
     WEIGHT KEYSTROKES FOR UNITS,
      TENS
66 V = V - 1: IF V > 0 THEN  VTAB
     6: HTAB 13: PRINT " ";: HTAB
     12: PRINT  INT (V / 18): GOTO
     60: REM  BLANK SECOND DIGIT
     OF SECONDS REMAINING
70 Z = 1: VTAB 16: HTAB 5: PRINT
      CHR$ (7);"TOO LATE!"
72  PRINT : PRINT : HTAB 3: INPUT
     "ANSWER? ";A
74  IF A = S THEN 78: REM CORRECT
      ANSWER
76 Z = 1: PRINT : HTAB 5: PRINT "
     WRONG, TRY AGAIN!": GOTO 72
78 P = P + 1: IF P = 51 THEN Z =
     1:Y = 6
80  PRINT : HTAB 5: PRINT "RIGHT!
     "
82  POKE 32, 0: IF Z = 1 THEN 21: REM
      Z STOPS GAME
84 X = X + 1: IF X > 9 THEN X = 0
     :Y = Y + 1:U = U - 2: REM CH
     ANGE ANIMAL AND TIME INTERVA
     L AFTER 10 PROBLEMS
86  GOTO 21
90  VTAB 24: INPUT "ANOTHER GAME
     (Y/N)? ";I$: IF I$ = "Y" THEN
      POKE 33, 40: HOME : GOTO 16
92  POKE 34, 0: POKE 33, 40: HOME :
      END
```

**ROADRUNNER**

50 ADD/SUBTRACT PROBLEMS

| | |
|---|---|
| NUMBER: 32 | SECONDS: 9 |
| SNAIL | 12 |
| TURTLE | — 4 |
| CHIPMONK | ANSWER? 7 |
| RABBIT | WRONG, TRY AGAIN. |
| COYOTE | ANSWER? 8 |
| ROADRUNNER | RIGHT! |

***Figure 1:*** **End of game, Problem 32 was answered incorrectly. "RABBIT" flashes to show highest level achieved.**

**Roadrunner**
**Variables List**

| | |
|---|---|
| A | Answer |
| C | Counter |
| | Delay |
| H | Highest Number |
| I | Input |
| I$ | Input |
| L | Top number for subtraction |
| M | Top number |
| N | Bottom number |
| P | Problem number |
| S | Sum or difference |
| S$ | Sign |
| T | Maximum time interval |
| U | Decreased time interval |
| V | Actual seconds remaining |
| W$(6) | Winning animal |
| W | Subscript |
| X | Counts ten problems |
| Y | Level attained |
| Z | Ends game |


# C.R.A.E.

A new Co-Resident Applesoft Editor for applesoft programmers. Now perform global changes/finds to anything in your applesoft programs. Quote range of lines from one part of your program to another. A fully optimized stop-list command that lists your program to the screen with no spaces added and forty columns wide. Append to other applesoft programs from disk. Get a formatted memory dump to help debugging. Renumber applesoft programs 5 times faster than most available renumber routines. Automatic line numbering. All utilities invoked with one-key commands.

C.R.A.E. changes your applesoft program right in memory and only needs to be loaded once.

Hardware requirements for all utilities are 48K Apple II or Apple II Plus, applesoft ROM and Disk.

C.R.A.E. 1....Includes global find/change, quote, dump and List renumber, autoline, and append.

Disk — $14.95

**Highlands Computer Services**
**14422 SE 132nd**
**Renton, WA 98055**

WA residents add 5.3% sales tax. Applesoft and Apple are registered trademarks of Apple Computers Inc.



# Southwestern Data Systems

presents

## NEW SOFTWARE BY ROGER WAGNER

**APPLE-DOC:** This program set is a must to anyone writing or using programs in Applesoft! It not only provides valuable information on each of your programs, but allows you to change any element throughout the listing almost as easily as you would change a single line! With APPLE-DOC you can produce a list of every variable in your program and the lines each is used on, each line called by a GOTO, GOSUB, etc., in fact, every occurance of almost anything! You can rename variables, even do local or global replacement editing on your listing.

DISKETTE OR CASSETTE: $24.95 (14 pgs. documentation)
(Applesoft only)

**ROGER'S EASEL:** What would be the easiest way to put lo-res graphics in a program? Probably to be able to just draw it with a sketch program and then somehow attach the picture itself to the program for later recall. Impossible? Not with ROGER'S EASEL! Create your own lo-res pictures with this program set, and then selectively attach them to your own Integer or Applesoft programs. Up to 40 pictures (or even text pages) can be permanently linked for nearly instantaneous (up to 10 pictures per second!) recall to either pg. 1 or 2 of text/graphics display. (Easel is in Int., Link prog. is in Int. & A/S)

DISKETTE OR CASSETTE: $16.95 (10 pgs. documentation)

**THE CORRESPONDENT:** At last an economical alternative to the higher priced text editors now available! THE CORRESPONDENT not only supports upper/lower case, but features character & line insert & delete functions. Output can be from 40 to 80 columns, with 4-directional scrolling to allow you to see exactly what will be output to the printer. It's very fast FIND routine makes it ideal as a free-form database for storing notes, phone lists, or anything where report generation is not essential. It will also access random or sequential text files — useful not only for the editing ability, but to examine & transfer data files, or for building your own EXEC files. routines are even included for putting the bidirectional scrolling in your own programs!

DISKETTE ONLY: $34.95 (32K Applesoft ROM or 48K RAM)

**PROGRAMMER'S UTILITY PACK:** A collection of many useful programs to aid in programming the Apple II. This set includes: Renumber-Applesoft & Int.: Printer output of old/new lines, non-destructive to mach. code within BASIC listing. Append-Applesoft & Int.: Easily join one program or section thereof to another. Address/Hex Converter: Converts all of the Apple's address formats including high- and low-order bytes for pointers. Line Find-Applesoft & Int.: Find the location of any BASIC line in memory for repairing garbaged programs, or inserting illegal statements like HIMEM:, CLR directly into listings. And there are many more! The extensive (14 pgs.) documentation alone is worth the price! It includes an in-depth explanation of the internal workings of Integer BASIC and Applesoft to allow you to do things you never thought possible on your APPLE!

DISKETTE OR CASSETTE: $16.95

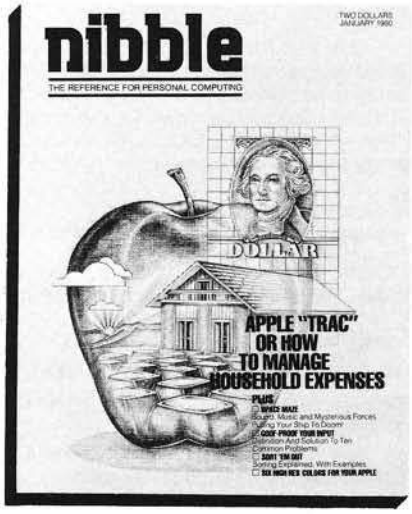
**(California residents add 6% Sales Tax)**

Available from your local computer store or:

**Southwestern Data Systems**
**P.O. Box 582-M**
**Santee, CA 92071**
**(714) 562-3670**

(Dealer inquiries invited)

# INTRODUCING . . . NIBBLE THE REFERENCE FOR APPLE COMPUTING




**NIBBLE IS:**
*A SOFTWARE GUIDE* for high quality Applications Programs for your Home and Business.

**NIBBLE IS:**
*A REFERENCE GUIDE* to new Programming Methods.

**NIBBLE IS:**
*A BUYERS GUIDE* for making purchase decisions on new products.

**NIBBLE IS:**
*A CONSTRUCTION PROJECT COOKBOOK* for adding function and value to the system you already own.

**NIBBLE IS:**
*A COMMUNICATIONS CLEARING HOUSE* for users, vendors, and associations.

Each issue of NIBBLE features at least one significant new application program of commercial quality. The programs in NIBBLE are surrounded with articles which show how to USE the programming methods in your OWN programs.

Examples of upcoming articles:
☐ Modeling and Forecasting Your Business ☐ Build a Two-Tape Controller for $12 ☐ Arcade Shooting Gallery — Save Your Quarters! ☐ Data Base Management System I, II, III

And many many more! NIBBLE will literally "Nibble Away" at the mysteries of your system to help you USE IT MORE. In 1980, the principal featured system is the Apple II.

**Try a NIBBLE**

**nibble**
BOX 325 Lincoln, Mass. 01773

**I'll try NIBBLE!**
**Enclosed is my $15 for 8 issues.**
**☐ check ☐ money order**

Name _____

Address _____

City _____

State _____ Zip _____

©1980 by micro-Software Publishing and Research Co., Lincoln, Mass. 01773. All rights reserved.
*Apple II is a registered trademark of Apple Computer Company

# Plotting with Special Character Graphics

**A primer on, and program for, generating plot mode type graphics with special characters. Applicable to the PET, Challenger, and other micros.**

**Dale DePriest**
**611 Galen**
**San Jose, CA 95106**

Microcomputers that support graphics are basically one of two types. One type supports their graphics with a special set of graphics characters that are printed or poked on the screen thereby drawing the picture you were trying to portray. Examples of this type of computer are the Challenger, Sorcerer, and Pet. The second type of graphics support divides the screen into small squares or rectangles which are turned on or off by specifying their address in a matrix system. These points are said to be "plotted" on the screen in the same fashion that you would plot on a piece of graph paper. Examples of this type of machine include the Compucolor, Intecolor and TRS-80.

If you are an owner of the first type of computer and have ever been envious of the people who own the second type or would like to use graphics programs written for the second type then this is the article for you! A program will be presented that allows you to duplicate the plot mode graphics and allows you to create your own expanded graphics mode.

First let us take a minute and ensure that everyone understands plot mode graphics. I will use the TRS-80 for an example. They have divided the screen into 128 points across (horizontally) and 48 points down (vertically). To identify any single point you must specify its location with two numbers; the first will identify how many points over from the left edge of the screen your point is and the second number will identify how may points down from the top edge. To turn a point on you would use the instruction SET (X,Y) where X is the distance across (0-127) and Y is the distance down (0-47). To draw a line across the screen you would have to write a program that would specify a value for Y and included a loop that would increment or decrement X until the line was drawn.

For example, consider the following program:

```
10 Y =6
20 FOR X =0 TO 63
30 SET (X,Y)
40 NEXT X
```

This program would plot a line near the top of the screen beginning at the left edge and extending across to the center of the screen.

The TRS-80 has three instructions that support their graphic; one to turn a point on, one to turn a point off, and one to examine a point to see whether it is on or off.

Some people call this graphics mode "high resolution" or simply HIRES graphics since this is the smallest increment of data that the programmer has control of. HIRES graphics capability of the TRS-80 is 128 × 48. A low resolution graphics capability exists also and is inherent in all computers whether they have graphics capability or not. This LOWRES graphics mode is simply to use a character form the keyboard (such as X) and draw pictures with it. Most people have used this technique when playing around with a typewriter. In this mode the TRS-80 would have a resolution of 64 × 16. (The number of characters per line by the number of lines.) To use this mode of graphics you would simply use print statements.

Intecolor and Compucolor use a method very much like that used by Radio Shack. They use a sequence of PLOT instructions to accomplish their graphics capability. In addition, they have some subplot modes which allow you to draw lines and bars without the necessity of writing loops like the one I showed you for the TRS-80. This capability gives the programmer a very powerful tool for fancy graphing such as vector mode, point to point lines, bar graphs, etc.

Most home computers display the screen from a memory somewhere in the 64K that a programmer has access to. The BASIC interpreter program simply places the information in the proper address of this memory and the hardware of the computer constantly displays this memory on the TV tube that you look at. For this reason, if you know where this refresh RAM (memory) is located, then you can simply use the BASIC PEEK instruction to find out what is there and the POKE instruction to put anything you choose on the screen, whether it be keyboard characters or graphics characters. Normally each character on the screen occupies one byte of data in the refresh RAM. This means that there are 256 possible characters that can be displayed. The way that each manufacturer uses these 256 characters is one of the major differences between home computers.

The TRS-80 is no exception. They reserve 64 of these characters for the keyboard, 64 of these characters are used for the double width characters and the other 128 are the graphics characters. Mathematically it can be shown that the 128 characters are enough to contain all possible combinations of 6 bits ($2^6 = 128$); therefore what Radio Shack did was divide a character into 6 pieces arranged in a two by three group.

Compucolor and Intecolor devote two bytes of data to each character on the screen since they must also include color information. This also gives then the ability to devote 256 characters to graphics, 8 bits in a two by four group, and thus they have a higher resolution capability than does Radio Shack.

Now that you understand plot mode character graphics let's see how we can duplicate this graphics mode with the special character graphics. The demonstration programs and the special graphics subroutine we'll be looking at will run unmodified on a PET computer but I'll try to include enough notes so that it should be easy to modify these routines for any computer that uses Microsoft BASIC as long as the graphics symbols are available.

The symbols that we are going to use are shown in Figure 1. Since we are going to divide our characters into four bits we will need $2^4$ or 16 characters. PET has thoughtfully provided all of these characters on the keyboard, although some will have to be used with the RVS key. Challenger doesn't provide these characters from the keyboard but you should be able to find them listed in the graphics program book. The first one is simply a space.

Since we plan to poke these characters on the screen, we'll need to know the decimal equilalents of all of these characters. The following subroutine will build an array of all the characters shown in figure 1 by using the decimal equivalents of these characters in the data statements.

```
32000  DIM X2 (15) :FOR Y1=0 TO
32010  READ X2(Y1):NEXT:RETURN
32020  DATA 32,126,124,226,123,97,
       225,236,108,127,225,251,
       98,252,254,160
```

If you don't have a PET, change the DATA statement per your machine documentation. Be sure to enter them in the order shown in figure 1.

My subroutines all use the variables X, Y, plus these variables with numbers. This is done to minimize the impact on variables you may be using in your program. Variable definitions are given in the table below.

X The horizontal coordinate of the point

Y The vertical coordinate ot the point

X1 The decimal address of the character that the point is in.

X2 The original data at the address X1

X3 A flag to tell the subroutine what kind of plotting is desired (see text)

Y1 The pointer into the array containing our plot character

Y2 A flag indicating which one of the four points in the character that X,Y points to

X2(Y1) The array of possible plot characters

Now let's look at the program in detail. Line 50 gets rid of any ambiguity about the value of X; first by making sure that it is an integer and then by making sure that the point is on the screen. The number 79 is one less than twice the number of characters you have across your screen. A good value to use on the Challenger 1P is 47; Challenger 2 would use 127.

Line 52 does the same thing for Y. The number 49 represents one less than twice the number of lines.

Line 54 generates the address of the character we are interested in and peeks the current value. It then searches to try and match this value with the array that we set up earlier. The number 40 is the number of characters on the Pet line. For a Challenger 1P this must be 32. 32768 is the decimal address of the starting location of the Pet memory map for screen refresh. Your system documentation should tell you where yours is located. For the Challenger 1P this starting location depends on the TV overscan but 53349 should be a good place to start.

Line 56. If the search is unsuccessful and X3 is a zero, we'll assume Y1 = 0, thereby overwriting any data that is already on the screen. Otherwise, we will abort the plot and preserve the data on the screen.

Lines 58 and 60 find the proper quadrant of the character.

Let's skip lines 63, 64 and 66 for now.

Line 68 does the actual plotting by oring the old data pointer and the quadrant pointer. If you've gotten this far and you suddenly find that your machine won't, or that there are two numbers together, then please drop me a line and I'll give you a program that does the same thing with logical IF tests. Be sure to tell me what kind of machine you have.

The program we have just discussed will simulate the TRS-80 SET instruction except that we can also have control over what happens should our plot program encounter a normal print character. To demonstrate this, consider the following example:

```
5 GOSUB 32000
10 Y = 6
20 FOR X = 0 TO 39
30 GOSUB 50
40 NEXT X : END
```

If you have entered the two subroutines prior to this, then this program will draw a line half way across the screen near the top, just like the Radio Shack program did. Now, remove these lines (5 through 40) and enter the following program.

```
10 PRINT CHR$(147):REM CLRS
SCREEN
20 GOSUB 32000
30 GOTO 100
100 FOR X = 0 TO 79
110 Y = 24 + 15 * SIN(X/5)
120 GOSUB 50
130 NEXT X
140 END
```

When you run this you should have a nice sine wave appear on the screen.

Radio Shack has a RESET instruction also that allows them to turn off a bit. Some of the time this is used to simulate a ball or bullet for animation purposes. Since this program is in Basic which is inherently slow for this sort of thing, I have provided for a special feature to allow simulating this kind of action. Please add this one line to your earlier program.

```
115 POKE X1, X2
```

This turns off the bit that was just turned on by poking back the original value. Once you have tried this program, please be sure to remove line 115. This is not the only use of the RESET instruction, however. We should be able to simulate this instruction also. Now we can discuss the rest of the main subroutine.

Line 62 holds the key to the power of this subroutine. By setting X3 to a particular value we can use this subroutine to do many plotting functions. We have already discussed the values of X3 = 0 and X3 = 1.

Line 64 is required since the ON instruction cannot work with X3 = 0.

Line 70 is the place we will jump to if X3 = 2. Add the following lines to your program and try it again.

```
140 IF X3 = 2 THEN END
150 X3 = 2
160 GOTO 100
```

As you can see, X3 = 2 simulates the RESET instruction very well. You should now save your program tape.

This same routine provides more advanced functions as well which are similar to those supplied by Compucolor. For example, if X3 = 3, then a test is made on the bit at the X,Y coordinate. If it is off, we'll turn it on but if it's already on, we'll turn it off. This decision is made with line 66.

Line 72 is the line we will get to if X3 = 4. This will cause us to enter the X-bar graph mode. Consider this program:

```
5 GOSUB 32000
10 PRINT CHR$(147)
```

```
15 Y = 6
20 X3 = 4
25 X = 39
30 GOSUB 50
35 END
```

It draws the same horizontal line we saw earlier but we didn't have to write a loop. It was also a little faster.

Line 74 is where you will end up when X3 = 5. This enters the Y-bar graph mode and vertical lines will be drawn.

Of course several more variations could be derived depending on your application. For example, if X3 = 6 pointed to line 76, then
76 GOSUB 68:X = X – 1:Y = Y + 1:GOTO 50
would draw a diagonal line. Or alternately,an adaptation of line 66 could provide a status that would indicate whether one bit was on or off.

### Final Considerations

This routine is intended to reside near the front of your using program since subroutines at the beginning of programs execute faster than those near the end. The initialization subroutine only executes once, so placing it at the end simply gets it out of the way.

### Notes on the Challenger 1P

In addition to the changes described in the text, you will have to make the following changes to run this program on the Challenger 1P. Line 54 will have to be broken into two lines since it is more than 72 characters long. The same is true of line 32020. Although Challenger has a very extensive set of graphics characters, they really blew it when using the sixteen characters described in this article. Four are missing. This prevents not only a clean implementation of this program but alos prevents another use for these characters such as that of creating large lettering. The best compromise may be the following statements:

32020 DATA 32, 168, 166, 155, 167, 156, 170, 175

32030 DATA 165, 169, 157, 177, 154, 178, 176, 161

Challenger should consider changing their ROM; perhaps changing 171 through 174.

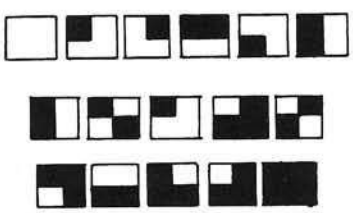
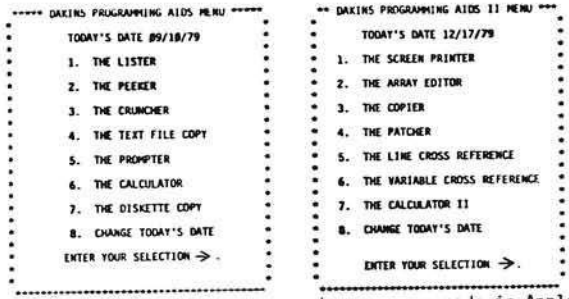


*Figure 1*

### Main Subroutine

```
50  X = INT(X): IF X  79 OR X  0 THEN RETURN
52  Y = INT(Y): IF Y  49 OR Y  0 THEN RETURN
54  X1 = INT(X/2) + 40*INT(Y/2) + 32768: X2 = PEEK(X1): FOR Y1 = 0 TO
    15: IF X2 = X2(Y1) THEN 58
56  NEXT: Y1 = 0: IF X3 THEN RETURN
58  Y2 = 1: IF X/2 – INT(X/2) THEN Y2 = 2*Y2
60  IF Y/2 – INT(Y/2) THEN Y2 = Y2*4
62  ON X3 GOTO 68,70,66,72,74
64  GOTO 68
66  IF X2(Y1 OR Y2) = X2(Y1) THEN 70
68  POKE X1, X2(Y1 OR Y2) : RETURN
70  POKE X1,X2(Y1 AND 15 – Y2) : RETURN
72  GOSUB 68 : X = X – 1 : GOTO 50
74  GOSUB 68 : Y = Y + 1 : GOTO 52
```


# DAKIN5 UTILITIES


***** DAKIN5 PROGRAMMING AIDS MENU *****
TODAY'S DATE 09/18/79
1. THE LISTER
2. THE PEEKER
3. THE CRUNCHER
4. THE TEXT FILE COPY
5. THE PROMPTER
6. THE CALCULATOR
7. THE DISKETTE COPY
8. CHANGE TODAY'S DATE
ENTER YOUR SELECTION →
** DAKIN5 PROGRAMMING AIDS II MENU ***
TODAY'S DATE 12/17/79
1. THE SCREEN PRINTER
2. THE ARRAY EDITOR
3. THE COPIER
4. THE PATCHER
5. THE LINE CROSS REFERENCE
6. THE VARIABLE CROSS REFERENCE
7. THE CALCULATOR II
8. CHANGE TODAY'S DATE
ENTER YOUR SELECTION →

The Cruncher removes REM statements and compresses code in Applesoft programs. The Prompter is a powerful data entry subroutine that can handle both string and numeric data. Options include using commas, decimal points, and leading zeros with right-justified numerics. A A maximum field length can be specified to prevent overflow in both numeric and alphanumeric fields. The Diskette Copy formats an output disk, copies each track, and verifies that the output matches the input. PLUS FOUR MORE UTILITIES TO AID YOUR OWN PROGRAMMING. Suggested Retail Price for Dakin5 Programming Aids is $39.95.

The Copier copies absolutely any kind of file or program from one diskette to another. The Variable Cross Reference produces a cross-reference for all variable names used in an Applesoft BASIC program. The Line Cross Reference creates a cross-reference for an Applesoft BASIC program, showing where a given line is referenced by GOTO, GOSUB, THEN, or LIST statements. The Patcher allows you to display any sector of a diskette, and then to update any data within that sector. PLUS THREE MORE UTILITIES TO FACILITATE YOUR OWN PROGRAMMING. Suggested Retail Price for Dakin5 Programming Aids II is $49.95.

Each utility package includes a program diskette and very complete documentation. The hardware requirements are an Apple II, 48K of memory, 2 Disk II's, and a printer. Languages are Applesoft/Assembler.

See your Apple dealer or contact Dakin5 Corporation, P.O. Box 21187, Denver, Colorado 80221. Telephone:(303) 426-6090. Dakin5 developed the business application software The Controller™ for Apple Computer Inc.

# Skyles Electric Works

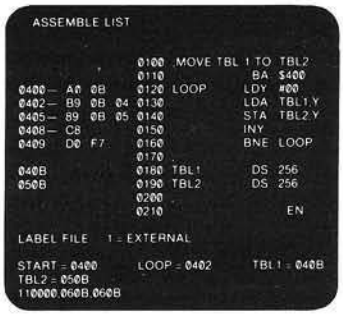
**Presenting the Skyles MacroTeA**

## The Software Development System For the Professional Programmer and The Serious Amateur Software Hobbyist

**...and for anyone who needs to automate dedicated industrial measurement and control applications.**

**The Macro TeA,** designed for use with the **Commodore PET** to create a remarkable synergism: a complete, integrated **software development system** for the 6502...the only 6502 system not requiring a separate disk drive. With over 60 commands for your complete machine language programming.

**...a lightning fast...fast...fast..... .....machine language assembler with true macro capabilities.** Assemble 16K source text in less than ten seconds! A single name indentifies a whole body of lines. Macro and conditional assembly support. Automatic line numbering. Labels up to ten characters long; $10^{16}$ different labels.




Install permanently without tools and in less time than it takes to load an equivalent sized assembler/text editor program from tape. No tape loading ever. And no occupying of RAM memory space: the MacroTeA puts 10K of executable machine language code in ROM (98ØØ to BFFF) and 2K in RAM (9ØØØ to 97FF).

**...a super powerful text editor.** 26 commands with string search and replace capability. Manuscript feature. Test loading and storage on tape or disks. Supports tape drives, disks, CRT, printers and keyboard.

**...an enhanced monitor with 11 powerful commands for program debugging and final polishing.**

**...with a warm-start button,** on a 12 inch cable. Reset the PET but not the 1792 bytes of object code in the **Macro TeA** RAM memory.

**...a completely solid state firmware system** ...all in ROM and RAM. No tapes to load. The system is available from the time you turn on your PET to the time you shut it off.

15 chips on a single high quality printed circuit board; interfaces with PET's parallel address and data bus or with Skyles Memory Adapter. A comprehensive 170 page manual is included

Truly, there is simply no other system of this magnitude at anywhere near this price. **$395.00** *

*(With any Skyles Memory Expansion System,* **$375.00**

**California residents: please add 6% or 6.5% sales tax as required*

VISA, MASTERCHARGE ORDERS CALL **(800) 538-3083** (except California residents)

CALIFORNIA ORDERS PLEASE CALL **(408) 257-9140**

**Skyles Electric Works** | **231 E South Whisman Road Mountain View, CA 94041 (408) 735-7891**

# SYM - 1 BASIC "GET" Command

**Everything you need to know to implement the 'GET' function in SYM - 1 BASIC. The use of the GET function is discussed and several examples are provided.**

**George Wells**
**1620 Victoria Place**
**La Verne, CA 91750**

The SYM-1 BASIC Interpreter provides for an unused "GET" token which always produces a Function Call error (FC) whenever it is encountered in a program. GET is an alternate form of INPUT except that it only inputs one character for each call and that one character can be any keyboard character including control characters and lower case letters. The first section of this article describes a simple procedure to implement this very useful command. The second section explains in detail how it works and the the third section offers some examples of BASIC subroutines utilizing the GET command.

### Section One
### Implementing the GET Command

Step 1: Deposit and Verify the code in the OBJECT LISTING. If it consistently will not Verify, read Section 2 before proceding.

Step 2: Enter the following monitor command:

```
.SD A600,A664
```

Step 3: Jump to BASIC:

```
.J O
```

Step 4: Enter and RUN a BASIC program such as:

```
100 PRINT "HIT ANY KEY:"
110 GET A$
120 PRINT ASC(A$)
130 GOTO 100
```

The GET command is always used to input a character string which will normally have a length of one. (A double-quote (") or a NULL results in a length of zero which causes an FC error to occur. See Section 2.) Of course, the string variable can be either simple or an element of a matrix, but only one variable is allowed for each GET and it cannot be used in a Direct Command. When GET is encountered in a running program there is no prompt "?" and prompt strings are not allowed. This is intentional to allow for several characters in a row to be typed in, in response to several GET's or for a loop which examines the characters for errors as they are typed. It is therefore normal to precede GET with a PRINT statement to serve as a prompt.

### Section Two
### Detailed Explanation of GET Implementation

The assembly language program to implement GET is stored in two sections of RAM that are unused by both the Monitor and BASIC. The first of these is the first 32 bytes of System RAM which are normally allocated as the Scope Buffer but are not changed in any way as long as none of the hex keypad buttons are pushed (except, of course, RST and DEBUG ON and OFF). These 32 bytes are located at $A600-$A61F. The second section of RAM is the 16 bytes located on page zero at $E8-$F7. The code can be entered into your SYM-1 and verified using the object code listing or if you have Synertek's RAE-1, you can enter the source code as it appears in the assembly listing. After it is assembled the block of code belonging on page zero must be moved there from page $0F with the monitor command:

```
B E8,FE8-FF7
```

The code can not be assembled directly on page zero since RAE-1 also uses that block of memory. If you happen to have EPROM in your system you can also relocate the code there (delete line 300 JMP GET.COMD.3). In order to activate GET, the System Output Vector ($A664,5) must be changed from its present value, assumed to be the Terminal Output routine (TOUT = $8AA0), to the GET command processor (GET.COMD = $A600). This vector can be changed at the monitor level with the simple command:

```
.SD A600,A664
```

or it can be done in BASIC with:

```
POKE 42596,0: POKE 42597,166
```

which can be either a Direct Command or part of a program. If you decide to relocate the code to some other address than $A600 then be sure to use the correct address when changing the System Output Vector. Please be aware of the fact that the System RAM is write protected after a warm start to BASIC (G O) until after a LOAD or SAVE command is attempted (if you have the new Monitor ROM) or until a call to ACCESS is made some other way, for example, with QQ = USR(&"8B86",0) or unless the jumper at 45-MM is removed. Incidentally, since BASIC passes the program size and file ID information to the Tape routines through the System RAM, the first LOAD or SAVE after a warm start won't work.

To understand how the GET command is processed look at the assembly language listing. Each time BASIC attempts to print any character, this routine will be entered. If the character to be printed is a carriage return, which is the case when any error is encountered, then further testing is performed to see if it is a Function Call error and then if it was caused by a GET token. If any of the proper conditions are not met then a jump is made to the Terminal Output routine or to whatever special output routine you might have.

Assuming that all the conditions for the GET command are met, then twelve bytes are taken off the stack to account for the series of JSR's involved in printing the error message. Next, the BASIC Input Buffer is set up as it would be if a single character were entered in response to an INPUT command. However, the routines that normally bring characters into the Input Buffer are bypassed because they ignore all control characters (except BELL) and change lower case letters to upper case. Instead, the Input Buffer is loaded directly by the GET command processor so that all characters will be allowed. In addition, a double-quote is automatically inserted before the typed character so that commas, colons and spaces will also be properly interpreted. After the typed character a zero is inserted which is the End-of-Line token. There remains an ambiguity over two characters which can be typed in, namely, NULL and double-quote ("), both of which will be interpreted as a string of zero length. The NULL looks like the End-of-Line token and the double-quote looks like the End-of-String character. If you are not concerned with this ambiguity in your application, skip the remainder of this section.

There are two ways to avoid the ambiguity between double-quote and NULL. First you can change the assembly language instruction on line 350 from AND #$7F to ORA #$80 and then subtract 128 from each character after the GET statement. Example: Change BASIC program line 630 to:

630 CHAR$ = CHR$(ASC(CHAR$)-128)

The second way to handle this is by inserting three instructions between lines 350 and 360 of the assembly program as follows:

```
CMP #$22
BNE +2
ORA #$80
```

But this will require relocating the code to accomodate the additional bytes of program. (Due to a minor error in RAE-1, the branch must be entered as BNE = +3.) In this case, only a double-quote has its most significant bit set. It is not necessary to subtract 128 as long as you treat the ASCII code for double-quote as 162 instead of 34. Also, line 630 of the BASIC program should be deleted.

### Section Three
### Examples of Using GET

The remainder of this article will describe several BASIC subroutines which can be used to simulate the INPUT function for integer, numeric and string variables. Also described is a means to disable the BREAK key to make it possible to write programs that are incapable of being clobbered by the operator. This is an especially important feature when running programs for the novice. If you've had the frustrating experience of trying to leave your computer in the hands of the kids to play games only to have them forget to press RETURN after every input and not press RETURN without some input, then you know what a boon this can be. It can save you from having to reload a program because the kids have unknowingly deleted lines of program by typing in numbers while in Command Level.

OBJECT LISTING

```
.V E8-F7
00E8 20 58 8A 29 7F 85 1F A2,F0
00F0 1D A0 00 84 20 4C EA C9,50
 0650
.V A600-A61F
A600 C9 0D D0 08 E0 08 D0 04,6A
A608 C0 36 F0 03 4C A0 8A BA,83
A610 8A 69 0B AA 9A A9 2C 85,1F
A618 1D A9 22 85 1E 4C E8 00,DE
 0DDE
.
```

```
LIST

 10 PROMPT$ = "INPUT A STRING: "
 11 GOSUB 600
 12 PRINT PHRASE$
 13 GOTO 10
 20 PROMPT$ = "INPUT A NUMBER: "
 21 GOSUB 500
 22 PRINT NUMBER
 23 GOTO 20
 30 PROMPT$ = "INPUT AN INTEGER: "
 31 GOSUB 400
 32 PRINT NUMBER%
 33 GOTO 30
 35 :
 95 REM  ••• SUBROUTINE TO ACTIVATE "GET" ROUTINE •••
100 QQ = USR(&"8B86",0): REM ALLOW ACCESS TO SYSTEM RAM
110 POKE 42596,0: POKE 42597,166: REM CHANGE OUTPUT VECTOR TO "GET"
120 RETURN
185 REM  ••• SUBROUTINE TO DISABLE "BREAK" KEY •••
195 REM SIMULATE MONITOR COMMAND: .SD 862D,A667
200 QQ = USR(&"8B86",0): REM ALLOW ACCESS TO SYSTEM RAM
210 POKE 42570,103: POKE 42571,166: REM STORE INSVEC+1 IN P3
220 POKE 42572,45: POKE 42573,134: REM STORE $862D (CLC-RTS) IN P2
230 QQ = USR(&"861D",0): REM EXECUTE STORE DOUBLE BYTE COMMAND
240 RETURN
285 REM  ••• SUBROUTINE TO ENABLE "BREAK" KEY •••
295 REM SIMULATE MONITOR COMMAND: .SD 8B3C,A667
300 QQ = USR(&"8B86",0): REM ALLOW ACCESS TO SYSTEM RAM
310 POKE 42570,103: POKE 42571,166: REM STORE INSVEC+1 IN P3
320 POKE 42572,60: POKE 42573,139: REM STORE $8B3C (TSTAT) IN P2
330 QQ = USR(&"861D",0): REM EXECUTE STORE DOUBLE BYTE COMMAND
340 RETURN
395 REM  ••• SUBROUTINE TO INPUT AN INTEGER •••
400 GOSUB 500: REM INPUT A NUMBER
410 IF ABS(NUMBER) > 32767 THEN 400: REM REPEAT IF OUT OF RANGE
420 NUMBER% = INT(ABS(NUMBER))*SGN(NUMBER): REM DROP FRACTIONAL PART
430 RETURN
495 REM  ••• SUBROUTINE TO INPUT A NUMBER •••
500 GOSUB 600: REM INPUT A STRING
510 NUMBER = VAL(PHRASE$): REM CONVERT STRING TO NUMBER
520 RETURN
595 REM  ••• SUBROUTINE TO INPUT A STRING •••
600 PRINT: PRINT PROMPT$;: REM PRINT PROMPT ON NEW LINE
610 PHRASE$ = "": REM DELETE PHRASE
620 GET CHAR$
630 IF LEN(CHAR$) = 0 THEN CHAR$ = CHR$(34): REM CHANGE NULL STRING TO "
640 IF ASC(CHAR$) <> 8 THEN 680: REM BRANCH IF NOT BACK-SPACE
650 IF LEN(PHRASE$) = 0 THEN PRINT RIGHT$(PROMPT$,1);: GOTO 620
660 PHRASE$ = LEFT$(PHRASE$,LEN(PHRASE$)-1): REM DELETE LAST CHARACTER
670 PRINT " "; CHR$(8);: GOTO 620
680 IF ASC(CHAR$) = 10 THEN 600: REM START OVER IF LINE-FEED
690 IF ASC(CHAR$) = 13 THEN PRINT: RETURN: REM DONE IF CARRIAGE RETURN
700 PHRASE$ = PHRASE$ + CHAR$
710 GOTO 620
795 REM  ••• SUBROUTINE TO DE-ACTIVATE "GET" ROUTINE •••
800 QQ = USR(&"8B86",0): REM ALLOW ACCESS TO SYSTEM RAM
810 POKE 42596,160: POKE 42597,138: REM CHANGE OUTPUT VECTOR TO "TOUT"
820 RETURN
OK
```

The BASIC program lisitng contains two parts. The first part (lines 10-35) contains sample drivers for the three types of INPUT's and the second part (lines 95-820) contains the actual subroutines. The first subroutine (GOSUB 100) changes the output vector to point to the assembly language program which of course must be loaded prior to entering BASIC. The last subroutine (GOSUB 800) can be used to switch the output vector back to its normal state. The second and third subroutines can be used to disable and enable the BREAK key. These routines use part of the Monitor Store Double Byte Command to change the Input Status Vector because it is impossible to do the same thing in pure BASIC since the status would be checked between the two POKE's and would result in the program going to an undesired place. The BREAK is disabled by simply pointing it to a routine that always returns a status clear.

The subroutine beginning at line 600 simulates the INPUT command for a character string. The first thing it does is print a prompt string which should be defined prior to calling the subroutine. The name of the prompt string is PROMPT$ (or PR$). Next, the string which will contain the typed characters is cleared. Its name is PHRASE$ (or PH$). Then a loop is entered which GETs the typed characters one at a time and examines them before it puts them into the PHRASE$ string to see if they are any of the following special characters:

1. NULL (same as double-quote) is changed to ".

2. Back Space deletes previous character.

3. Line Feed deletes entire line.

4. Carriage Return ends the input.

No test is made to limit the number of characters to 255. Therefore, typing in 256 characters is a way to "BREAK" a program that has the BREAK key disabled since it will cause a Long String Error (LS).

The subroutine beginning at line 500 simulates the INPUT command for a number. It does this by calling the string input subroutine and using the BASIC VAL function to put the string into the variable called NUMBER (or NU). If the string does not convert correctly into a number, no error is generated, instead that portion of the string up to the error is used (or zero if it is completely wrong). However, if the magnitude of the number is too large for BASIC an Overflow Error(OV) results. This is another way to "BREAK" a program even with the BREAK key disabled.

The subroutine beginning at line 400 simulates the INPUT command for an integer. It does this by calling the number input subroutine and using the BASIC INT function to convert it to an integer called NUMBER• (or NU•). If the number is too large to be an integer, the prompt is repeated to avoid an error. Also, the fractional part of a negative number is dropped instead of rounding up to the next larger integer (absolute value).

Obviously, similar sorts of routines can be written to accomodate any particular requirements you might have. One word of caution: at the lower baud rates BASIC can't keep up with a fast typist. Using the BREAK disable subroutine will keep the program from aborting but might result in incorrect characters being read. However, if they are read incorrectly they will also be echoed incorrectly, so backspace over any errors and retype. At 4800 baud, BASIC can easily keep up with all but the fastest typist. At 110 baud it isn't hard to get incorrect reads, but even then it's not likely to be a problem with a novice operator. However, if you are running at 110 baud it is probably because you are running on a teletype in which case you will have to handle the character deletes with something other than a back-space.

```
>ASSEMBLY LISTING

                0010 GET.TOKEN   .DE $9B       BASIC "GET" TOKEN
                0020 FC.ERROR    .DE $08       BASIC "FC ERROR" TOKEN
                0030 INPUT.COMD  .DE $C9B9     BASIC INPUT COMMAND INTERPRETER
                0040 INP.BUFFER  .DE $1E       BASIC INPUT BUFFER
                0050 TOUT        .DE $8AA0     MONITOR TERMINAL OUTPUT ROUTINE
                0060 INTCHR      .DE $8A58     MONITOR INPUT TERMINAL CHARACTER
                0070
                0080             .OS
                0090             .BA $A600
                0100
                0110 ;   ••• PROGRAM TO IMPLEMENT SYM-1 BASIC "GET" COMMAND •••
                0120
A600- C9 0D     0130 GET.COMD    CMP #$0D              TEST FOR CARRIAGE RETURN
A602- D0 08     0140             BNE GET.COMD.1        AND BRANCH IF NOT.
A604- E0 08     0150             CPX #FC.ERROR         TEST FOR FC ERROR AND
A606- D0 04     0160             BNE GET.COMD.1        BRANCH IF NOT.
A608- C0 36     0170             CPY #L,GET.TOKEN+GET.TOKEN    TEST FOR GET AND
A60A- F0 03     0180             BEQ GET.COMD.2                BRANCH IF SO.
A60C- 4C A0 8A  0190 GET.COMD.1  JMP TOUT              IF NOT, CONTINUE OUTPUT.
                0200
A60F- BA        0210 GET.COMD.2  TSX                   TAKE 12 BYTES OFF STACK.
A610- 8A        0220             TXA                   ALREADY IN BINARY MODE AND
A611- 69 0B     0230             ADC #12-1             CARRY SET, SO ADD 11.
A613- AA        0240             TAX
A614- 9A        0250             TXS
A615- A9 2C     0260             LDA #',               STORE COMMA IN FRONT OF
A617- 85 1D     0270             STA •INP.BUFFER-1     BUFFER (NEEDED BY BASIC).
A619- A9 22     0280             LDA #'"               STORE QUOTE IN BUFFER TO
A61B- 85 1E     0290             STA •INP.BUFFER       ALLOW AUTO STRING INPUT.
A61D- 4C E8 00  0300             JMP GET.COMD.3        CONTINUE ON PAGE ZERO.
                0310
                0320             .BA $E8               STORE $E8 CODE AT $FE8,
                0330             .MC $FE8              MOVE WITH:  B E8,FE8-FF7
00E8- 20 58 8A  0340 GET.COMD.3  JSR INTCHR            INPUT A CHARACTER.
00EB- 29 7F     0350             AND #$7F              CLEAR PARITY BIT.
00ED- 85 1F     0360             STA •INP.BUFFER+1     PUT IT IN BUFFER.
00EF- A2 1D     0370             LDX #INP.BUFFER-1     X NEEDED BY BASIC.
00F1- A0 00     0380             LDY #0                Y=0 NEEDED BY BASIC.
00F3- 84 20     0390             STY •INP.BUFFER+2     END-OF-LINE TOKEN.
00F5- 4C EA C9  0400             JMP INPUT.COMD+49     CONT INTO BASIC.
                0410             .EN

LABEL FILE:  [ / = EXTERNAL ]

/GET.TOKEN=009B          /FC.ERROR=0008          /INPUT.COMD=C9B9
/INP.BUFFER=001E         /TOUT=8AA0              /INTCHR=8A58
GET.COMD=A600            GET.COMD.1=A60C         GET.COMD.2=A60F
GET.COMD.3=00E8
//0000,00F8,0FF8
>
```


## Classified Ads

INTEGER PASCAL FOR APPLE II
Translator produces 6502 code from P-codes, 4X faster than Integer BASIC. Compiler, interpreter: $30. Translator additional $35. Requires 48K and disk. Cal. res add 6%

M&M Software Co.
380 N. Armando, #Z-19
Anaheim, CA 92806

SYM-PHYSIS
The SYM-1 Users' Group Newsletter Software/Firmware/Hardware $9.00 ($12.50 overseas) for 6 issues.
P.O. Box 315
Chico, CA 95927
(916) 895-8751

# EXCERT, INCORPORATED

## • • • AIM - 65 • • •

### SPECIAL

**A65-4AB** — **AIM-65 w/4K RAM, Assembler and BASIC ROM** — **$595**

### SPARE PARTS (When Available)

| P/N | | QTY 1 - 9 |
|---|---|---|
| A65-1 | AIM-65 w/1K RAM | $375 |
| A65-4 | AIM-65 w/4K RAM | $450 |
| A65-A | Assembler ROM | $85 |
| A65-B | BASIC ROM | $100 |
| A65-P | Printer | $40 |
| A65-D | Complete Display Board | $85 |
| | w/Exchange of Old Board | $50 |
| A65-K | Keyboard | $30 |

### ACCESSORIES

| P/N | | QTY 1 - 9 |
|---|---|---|
| **Power Supplies** (Fully AIM-65 Compatible) | | |
| PRS3 | +5V at 3A, +24V at 1A w/mtg hardware, cord, etc. | $65 |
| PRS4 | +5V at 2A, +24V at .5A w/mtg hardware, cord, etc. | $50 |
| PR55 | +5V at 2A, +24V ±15% at .5A ±12V to ±15V at .4A | $75 |
| **From The Enclosures Group** | | |
| ENC1 | AIM-65 case w/space for PRS3/PRS4 | $45 |
| ENC1A | AIM-65 case w/space for PRS3/PRS4 and one expansion board | $49 |
| **Cases with Power Supplies** | | |
| ENC3 | ENC1 w/PRS3 mounted inside | $115 |
| ENC3A | ENC1A w/PRS3 mounted inside | $119 |
| ENC4 | ENC1 w/PRS4 mounted inside | $100 |
| ENC4A | ENC1A w/PRS4 mounted inside | $104 |
| ENC5 | ENC1 w/PRS5 Mounted inside | $125 |
| ENC5A | ENC1A w/PRS5 Mounted inside | $129 |
| **From The Computerist, Inc.** | | |
| MCP1 | Mother Plus™ Dual 44 pin mother card takes MEB1, VIB1, PTC1, fully buffered, 5 expansion slots underneath the AIM | $80 |
| MEB1 | Memory Plus™ 8K RAM, 8K Prom sockets, 6522 I/O chip and programmer for 5V EPROMS (with cables $215) | $200 |
| PTC1 | Proto Plus™ Prototype card same size as KIM-1, MEB1, VIB1 | $40 |
| VIB1 | Video Plus™ board with 128 char, 128 user char, up to 4K display RAM, light pen and ASCII keyboard interfaces w/cables | $245 |
| CCP1 | Cage Plus™ mounts to Mother Plus™ w/ card guides | $25 |

| P/N | | QTY 1 - 9 |
|---|---|---|
| **From Optimal Technology** | | |
| ADC1 | A/D eight channels, D/A 2 channels. Requires ±12V to ±15V at 100MA & 2-I/O Ports w/cnnctr from AIM-6522 | $115 |
| **From Seawell Marketing, Inc.** | | |
| MCP2 | Little Buffered Mother™ Single 44 pin (KIM-4 style) mother card takes MEB2,PGR2, PTC2 and PIO2. Has on board 5V regulator for AIM-65, 4 expansion slots. Routes A&E signals to duplicates on sides | $139 |
| | with 4K RAM | $189 |
| MEB2 | SEA 16™ 16K static RAM board takes 2114L with regulators and address switches 16K | $199 |
| PGR2 | Prommer™ Programmer for 5V EPROMS with ROM firmware, regulators, 4 textool sockets, up to 8 EPROMS simultanously, can execute after programming | $299 |
| PIO2 | Parallel I/O board with 4-6522's | $260 |
| PTC2 | Proto/Blank™ Prototype card that fits MCP2 | $49 |
| PTC2A | Proto/Pop™ with regulator, decoders, switches | $99 |
| **From Beta Computer** (Close Out—Limited Qty) | | |
| MEB3 | 32K Dynamic Memory Card w/on board DC to DC converters (5V only .8Amax) | $375 |
| **Miscellaneous** | | |
| TPT2 | Approved Thermal Paper Tape 5/165' rolls | $10 |
| MEM6 | 6/2114 RAM Chips | $45 |

### SYSTEMS

**We specialize in assembled and tested systems made from the above items. Normally, the price will be the total of the items, plus $5 for shipping, insurance and handling. Please call or write for exact prices or if questions arise. Six month warranty on all systems.**

Mail Check or Money Order To:


**EXCERT, INC.**
**P.O. Box 8600**
**White Bear Lake, MN 55110**
**(612) 426-4114**


Higher quantities quoted upon request.
COD's accepted.
Add $5.00 for shipping, insurance and handling.
Minnesota residents add 4 % sales tax.
Prices subject to change without notice

# A Simple Temperature Measurement Program and Interface

**Using a micro for temperature measurement demonstrates some of the problems and some of the solutions involved in interfacing to the real world.**

**Marvin L. DeJong**
**Dept. of Math & Physics**
**The School of the Ozarks**
**Point Lookout, MO 65726**

Temperature measurements at least as precise as +1°C can be made with the circuit shown in Figure 1 and the program listed in Table 1. The 555 timer integrated circuit operates in conjunction with a FENWAL GB41P2 thermistor as a temperature-to-frequency converter. The pulses from the circuit in Figure 1 are counted with the T2 counter/timer on the 6522 Versatile Interface Adapter. A machine language subroutine measures the number of pulses in one second, while a BASIC program converts the frequency to temperature.

The relationship of the temperature of the thermistor to the frquency of the puslses at PB6 is non-linear. A temperature -Vs- frequency curve for our system is shown in Figure 2. You must make such a clibration curve for the system to work. A calibration curve is obtained by immersing the thermistor and a previously calibrated thermometer in some fluid and making measurements of the frequency as the temperature of the fluid is changed. We used water, heat, and ice cubes to produce our calibration curve. The frequency measurement program in Table 2 is used to measure the pulse frequency as a function of temperature. If you want to use this system as an air themometer, then the fluid should be air. You will have to wait for nature to provide the necessary temperature changes. Temperatures below and above those shown on our calibration curve (Figure 2) may be included, depending on your intended application. Provided components with low temperature coefficients are used in the 555 timer circuit, the precision of the temperature measurements made by the program will depend largely on the quality of the calibration data you obtain for your circuit. The thermistor may be located in some remote location and connected to the 555 timer circuit by a twisted wire pair.

The program listed in Table 1 requires the user to input 20 frequency -temperature points from the calibration curve. The program can be easily modified to input more or less data. With the calibration data in memory, it calls the machine language subroutine to measure the frequency of the pulses from the interface circuit in Figure 1. Using the measured frequency and the calibration data, it performs a quadratic interpolation calculation to conver the frequency measurement to a temperature. It also converts the Celsius temperature to a Fahrenheit temperature and outputs both. In the BASIC program, statements number 50, 60 and 70 serve to get the frequency using the machine language subroutine. We are using AIM 65 BASIC, and the techniques necessary to call the machine language subroutine may vary from machine to machine. In any case,

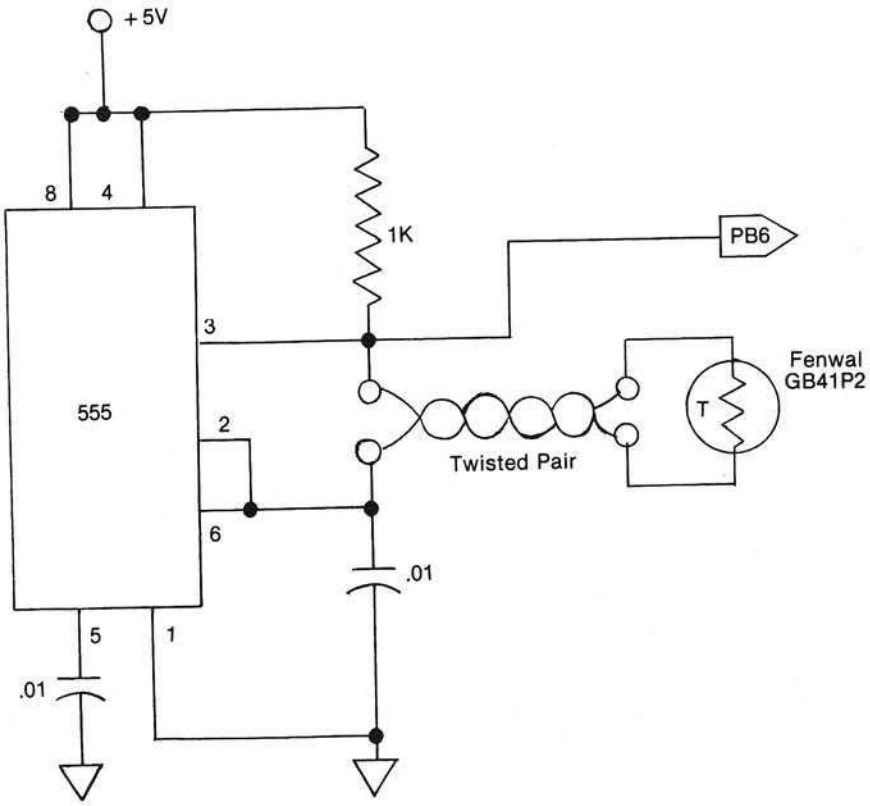



***Figure 1:*** **Using the 555 Timer as a Temperature-to-Frequency Converter**

statement number 50 pokes the starting address of th machine language subroutine into a location where AIM 65 BASIC can find it. Statement number 60 actually produces the subroutine jump. The variable Y means nothing in statement 60. In statement 70 the BASIC program obtains the frequency from the two bytes in memory where the machine language subroutine stored it, namely 49 = $31 and 50 = $32 in zero page.

Because of the way the quadratic interpolation formula is applied to the incoming frequency data, it is a good idea to make the first calibration point entered into the BASIC program be F = 0, T = −100 or some other low temperature below the range where you wish to operate. The other calibration points, from your calibration curve, are entered **in order** from low frequency-low temperature to high frequency-high temperature. For example, our first few data points entered were:

0, −40
1000, −10
2550, 0.5
3000, 5.0

A close inspection of our calibration curve in Figure 2 shows that the first two sets of points are a dummy point (0, −40) and an extrapolated point (1000, −10). Note that the data are entered in pairs, frequency first, temperature second.

For reference purposes, let's review very briefly the quadratic interpolation formula that is used. Given a function T(F) defined at three points, $(F^i, T^i)$, $(F^j, T^j)$, and $(F^k, T^k)$, we must find the value of the function at an arbitrary point F, assuming that the curve through the three points is a second degree equation (quadratic) in F. The equation is:

$$T = T_j + U\,[-R^2T_i + (R^2 - L^2)T_j + L^2T_k] + U^2\,\frac{(R+L)}{RL}\,[RT_i - (R+L)T_j + LT_k],$$

where, $R = F_j - F_i$, $L = F_k - F_j$, and $U = (F - F_j)/(R + L)$.

Refer to Figure 3 for a graphical interpretation of quadratic interpolation. In the program, the value of j (J in BASIC) that is chosen is such that F exceeds $F^j$ but is less than $F^k$. Then i = j − 1 and k = j + 1. Thus the points $F^j$ and $F^k$ always bracket F.

Now a few comments on the machine language subroutines used in the programs in Tables 1 and 2. These routines are identical. They allow the T2 counter/timer on the 6522 to count pulses for a number of 50,000 clock cycle intervals. The number of such intervals is determined by the byte of data in location $OF07 in the program. $14 = 20 such intervals give a total counting period of one second. Clearly this number may be changed to count pulses for either 0.1 s, 10.0 s, or some other time interval if

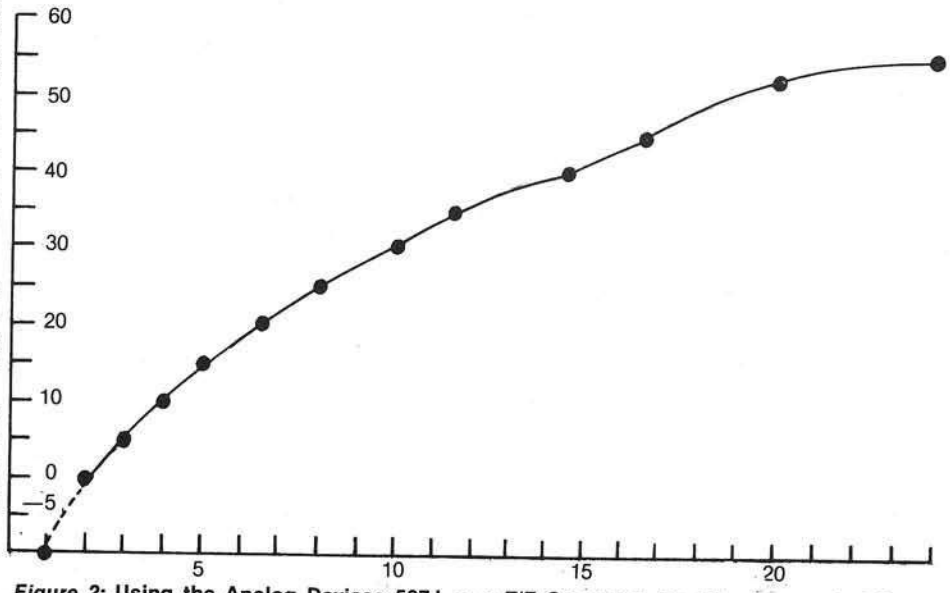



***Figure 2:* Using the Analog Devices 537J as a T/F Converter. Resistances are in Ohms and Capitances are in Microfarads.**

Table 1 . A simple frequency-to-temperature conversion program.

```
10 DIM F(20), T(20)
20 FOR J = 1 TO 20
30 INPUT F(J),T(J)
40 NEXT J
50 POKE Ø4,ØØ: POKE Ø5, 15
60 Y=USR(Ø)
70 Y=PEEK(49): Z=PEEK(50)
80 FRQ=256*Z+Y
90 FOR J = 1 TO 20
100 IF FRQ <F(J) THEN 120
110 NEXT J
120 I=J-1: K=J+1
130 L=F(J) - F(I)
140 R=F(K) - F(J)
150 U=(FRQ - F(J))/(R + L)
160 TC=T(J) + U/(R*L)*(-R*R*T(I)+(R*R-L*L)*T(J)+L*L*T(K))
170 TC=TC+U*U*(R+L)/(L*R)*(R*T(I)-(R+L)*T(J)+L*T(K))
180 TF = 9/5*TC + 32
190 TC= INT(TC + .5): TF= INT(TF + .5)
200 PRINT " ";TF;"F ";TC;"C"
210 GO TO 60
220 END
```

```
$OF00 D8        START   CLD          $OF1C 2C 0D A0   TEST    BIT IFR
 OF01 A9 60             LDA $60       OF1F 50 FB              BVC TEST
 OF03 8D 0B A0          STA ACR       OF21 AD 04 A0           LDA T1CL
 OF06 A9 4D             LDA $4D       OF24 C6 30              DEC CNTR
 OF08 8D 06 A0          STA T1LL      OF26 D0 F4              BNE TEST
 OF0B A9 14             LDA $14       OF28 38                 SEC
 OF0D 85 30             STA CNTR      OF29 A9 FF              LDA $FF
 OF0F A9 C3             LDA $C3       OF2B ED 08 A0           SBC T2CL
 OF11 8D 05 A0          STA T1LH      OF2E 85 31              STA PLSLO
 OF14 A9 FF             LDA $FF       OF30 A9 FF              LDA $FF
 OF16 8D 08 A0          STA T2LL      OF32 ED 09 A0           SBC T2CH
 OF19 8D 09 A0          STA T2CH      OF35 85 32              STA PLSHI
                                      OF37 4C D1 C0           JMP BASIC*
```

*Used in AIM 65 BASIC. Other BASICs may use a different return-from-subroutine technique.

Table 2. A program to measure frequency using BASIC and machine language.

```
10 POKE Ø4,ØØ: POKE Ø5, 15
20 Y = USR(Ø)
30 FRQ = 256*PEEK(5Ø) + PEEK(49)
40 PRINT FRQ
50 GO TO 20
```

```
$0F00 D8          START   CLD           Clear decimal mode.
 0F01 A9 60               LDA $60       Set up ACR so T1 runs free and
 0F03 8D 0B A0            STA ACR       T2 counts pulses.
 0F06 A9 14               LDA $14       The program will count pulses for
 0F08 85 30               STA COUNT     $14 = 20 intervals of 50,000 clock
 0F0A A9 4D               LDA $4D       cycles.  T1 is loaded with $C34D,
 0F0C 8D 06 A0            STA T1LL      since $C34D + 2 = 50,000.  IFR6 will
 0F0F A9 C3               LDA $C3       be set every 50,000 clock cycles.
 0F11 8D 05 A0            STA T1LH      Clear IFR6 and start T1 running.
 0F14 A9 FF               LDA $FF       Set up T2 to start counting down
 0F16 8D 08 A0            STA T2LL      from $FFFF.
 0F19 8D 09 A0            STA T2CH      Start counting pulses on PB6.
 0F1C 2C 0D A0     LOAF    BIT IFR       Has T1 timed out yet?
 0F1F 50 FB               BVC LOAF      No, then wait in this loop.
 0F21 AD 04 A0            LDA T1CL      Read T1CL simply to clear IFR6.
 0F24 C6 30               DEC COUNT     Decrement interval counter.
 0F26 D0 F4               BNE LOAF      Count pulses for another interval if
 0F28 38                  SEC           interval counter has not reached zero.
 0F29 A9 FF               LDA $FF       If it has reached zero, obtain the
 0F2B ED 08 A0            SBC T2CL      number of pulses from T2 by subtracting
 0F2E 85 31               STA PULSLO    the number in T2 from $FFFF.
 0F30 A9 FF               LDA $FF       Result into locations $0031 and $0032.
 0F32 ED 09 A0            SBC T2CH
 0F35 85 32               STA PULSHI
 0F37 4C D1 C0            JMP BASIC     AIM 65 return to BASIC command.
```

necessary. The programs in Tables 1 and 2 will count to a maximum of 65535 pulses in one one-second interval at a maximum rate of 500,000 Hz, the limit of the 6522. Note that the total counting interval ma be more or less than one second by say ten microseconds. This error amounts to less than one count if the incoming pulse rate is less than 65,535 Hz, and is of no consequence for this application. The listing in Table 2 is useful as a frequency counter with no regard to our frequency-to-temperature conversion program listed in Table 1. That is, the program in Table 2 is a stand-alone frequency counting program which may be used to count the frequency of pulses arriving at PB6, provided these are TTL level pulses similar to those provided by the 555 temperature-to-frequency circuit. A clever programmer will note that if IFR5, the T2 interrupt flag, is read, the T2 counter becomes a 17 bit counter, extending the range listed above by a factor of two. We did not program this feature into the programs in Tables 1 and 2.

Now that you can measure temperature, let's see what interesting applications you can come up with, and please let us hear from you. Of course, the first thing you will want to do is put the thermistor under your tongue and measure your body temperature. Analog Devices sells a T/F converter (AS537) that provides a linear relationship between T and F. We now describe how to interface it to your computer.

The connection diagram for the AD537 is shown in Figure 4. Again, the T2 timer/counter on the 6522 is used to measure the frequency of the pulses coming from

### *A Program to Measure Temperature with the AD537 Interface*

```
10 POKE Ø4,ØØ: POKE Ø5,15
20 Y = USR(Ø)
30 FRQ = 256*PEEK(5Ø) + PEEK(49)
40 TC = (FRQ - 2731)/1Ø
50 TF = TC*9/5 + 32
60 TF = INT(TF + .5)
70 PRINT " "; TF; "F"; TC; "C"
80 GO TO 2Ø
```

the AD537. With the values shown, the AD537 will produce a linear relationship between frequency and absolute temperature (Kelvin degrees) of 10Hz/°K. At room temperature (about 300°K) the frequency will be 3000 Hz. The 15 k potentiometer in Figure 4 is adjusted to give the correct temperature. The adjustments are easier if the 15 k potentiometer is replaced by a 9.1 k resistor in series with a 2 k potentiometer to trim the total resistance to about 10 k ohms.

To convert from absolute temperature (°K) to Celsius temperature, we make use of the formula [°C = °K − 273.1]. Then we can convert to Fahrenheit with the formula [°F = (°C)(9/5) + 32]. The entire process is handled with the BASIC program listed in Table 3. This program also calls the machine subroutine listed in Tables 1 and 2.

The AD537 is a versatile device. It can also be used as a very fine voltage-to-frequency converter with only a few external components. Analog Devices appears to share my philosophy that the fewer external components around, the less likely it is for me to have problems. In any case, with the same integrated circuit you can make youself a voltmeter. The same machine language subroutine will provide the necessary software, and a simple BASIC calling program will place the decimal point and output the answer. You should be sure to obtain the specification sheets on the AD537 if you get one. They contain a lot of useful and vital information. For example, the Ad537 can be operated in a remote location with a two-wire connection. Several of them can be multiplexed because the pulse output pin is an open-collector connection. The AD537 is much more expensive than a 555 timer, and jAnalog Devices may require a minimum order. Perhaps the members of your computer club can get together and place an order. Write: *Analog Devices*, 1 Industrial Park, P.O. Box 280, Norwood, Ma. 02062.

If you do not have a BASIC interpreter on your computer, then the machine language output subroutines given in Tables 4, 5, and 6 may be used with the programs in Tables 2 and 3 to output the frequency and temperature information. (Note that in order to measure temperature with the 555 timer circuit, a BASIC interpreter is an absolute essential.) SYM-1 and KIM-1 users can use the binary-to-BCD conversion routine in Table 4, together with their own subroutine that displays six numbers on the 7-segment LEDs, to display the frequency of the pulses measured by the machine language program in Table 2. The JMP BASIC instruction at $0F37 would be replaced by the following instructions:

```
20 60 OF      JSR DCML
20 ?? ??      JSR DISPLAY
4C 00 OF      JMP START
```

where DISPLAY is the user's routine to display six digits. AIM 65 owners will want to use all the subroutines in Tables 4, 5, and 6 to output the BCD digits representing the frequency to the AIM 65 twenty character display. To use the subroutines with the AD537 interface and the program in Table 3, you must first subtract $0AAB (2731) from the measured pulse rate to convert it to Celsius, and then output the BCD digits, remembering for yourself where the decimal place is. Good luck.

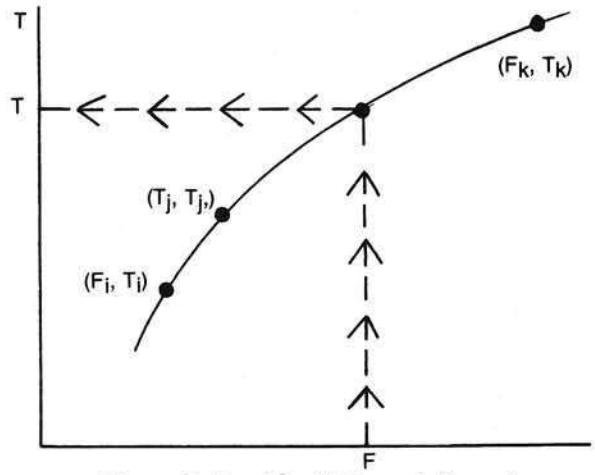



***Figure 3:* Graphical Interpretation of Quadratic Interpolation**

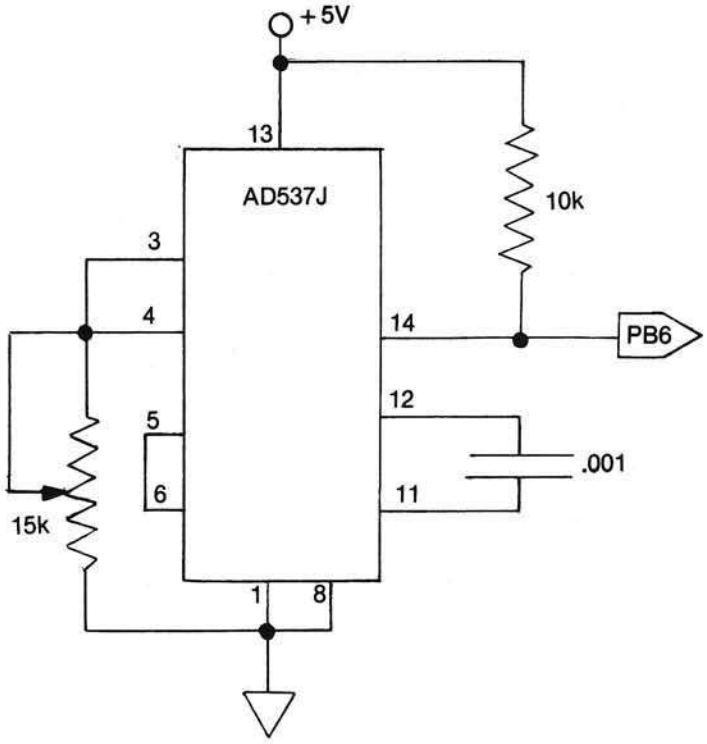



***Figure 4:* Frequency-to-Temperature Conversion Curve for the 555 Circuit.**

Table 4. A subroutine to convert a 16-bit binary number to six BCD digits.

```
$0031 = PLSLO; contains low-order byte of 16-bit number to be converted.
$0032 = PLSHI; contains high-order byte of 16-bit number to be converted.
$0F60 A9 00      DCML    LDA $00          Clear memory locations for the
 0F62 85 01              STA BCDLO        BCD number
 0F64 85 02              STA BCDMI
 0F66 85 03              STA BCDHI
 0F68 F8                 SED              Set decimal mode for subsequent
 0F69 A0 10              LDY $10          additions.  Y contains number of
 0F6B 06 31      THERE   ASL PLSMO        bits to be converted.  One bit of
 0F6D 26 32              ROL PLSMI        the number is shifted into the
 0F6F A2 FD              LDX $FD          carry bit at a time.  X serves as
 0F71 B5 04      MORE    LDA BYT,X        a counter for a triple-precision
 0F73 75 04              ADC BYT,X        addition.
 0F75 95 04              STA BYT,X
 0F77 E8                 INX
 0F78 D0 F7              BNE MORE
 0F7A 88                 DEY              Get the next bit.
 0F7B D0 EE              BNE THERE        When Y = 0, the conversion is
 0F7D D8                 CLD              complete.
 0F7E 60                 RTS              Return to calling program.
```


**Would You Like to Become a MICRO Dealer?**

MICRO is a quality 6502 magazine. Our current dealers report that having MICRO available for sale in their store actually helps sell 6502 based systems.

We require a minimum quantity of only 10 copies per month and we offer a standard trade discount.

Other items available are the Best of MICRO, vols 1 and 2, and All of MICRO.

If interested, please write to:

MICRO
P.O.Box 6502
Chelmsford, MA 01824

Table 5. A subroutine to convert six BCD digits to ASCII and call an output subroutine.

```
$0F80 A2 06     ASCII   LDX $06     X contains the number of BCD digits.
 0F82 A9 00             LDA $00     Our out-character (OUTCH) subroutine
 0F84 85 04             STA LOC     requires LOC to start at $00.
 0F86 A5 03     NEXT    LDA BCDHI   Get the most-significant nibble
 0F88 29 F0             AND $F0     of the BCD number.  The BCD digits
 0F8A 4A                LSR A       will be output from the most-
 0F8B 4A                LSR A       significant to the least significant.
 0F8C 4A                LSR A       Move high-order nibble to the low-
 0F8D 4A                LSR A       order nibble.
 0F8E 18                CLC
 0F8F 69 30             ADC $30     Adding $30 to a BCD digit converts
 0F91 20 A5 0F          JSR OUTCH   it to ASCII.  Output the character.
 0F94 A0 04             LDY $04     Get another nibble.
 0F96 06 01     AGN     ASL BCDLO
 0F98 26 02             ROL BCDMI
 0F9A 26 03             ROL BCDHI
 0F9C 88                DEY
 0F9D D0 F7             BNE AGN
 0F9F CA                DEX         Get another digit?
 0FA0 D0 E4             BNE NEXT    Yes.
 0FA2 60                RTS         No.
```

Table 6. A subroutine to display six digits on the AIM 65 display.

```
$0FA5 09 80     OUTCH   ORA $80     ASCII character is in the accumulator.
$0FA7 85 05             STA TEMP    Set bit seven and store temporarily.
$0FA9 8A                TXA         Save X.
$0FAA 48                PHA
$0FAB A6 04             LDX LOC     LOC contains location of the digit
$0FAD A5 05             LDA TEMP    on the 20 character display
$0FAF 20 7B EF          JSR OUTDD1  Use AIM 65 monitor routine.
$0FB2 E6 04             INC LOC
$0FB4 A5 04             LDA LOC
$0FB6 C9 06             CMP $06     Have all six characters been output?
$0FB8 90 04             BCC AHEAD
$0FBA A9 00             LDA $00     Yes.  Clear LOC.
$0FBC 85 04             STA LOC
$0FBE 68        AHEAD   PLA         Get X back.
$0FBF AA                TAX
$0FC0 60                RTS         Return to the calling routine.
```


## Advertise With Us

We have just put together a new Media Package. It includes all that you need to know about rates, sizes, and placement.

Also included is information on color ads, which will be available next month, in our June issue. We give you all the specifications that you need to know about negatives, color separation, insertions, etc.

We also give you information on contracts and ad placement, typesetting, and artwork.

Write today for our new Media Package. The address is:

Advertising Dept
MICRO
Box 6502
Chelmsford, MA 01824

or call and ask for Terry, Cathi, or Mary. All three are ready to help you design a sharp looking ad. Bring your ad to us, and we'll do the rest to bring it to the world of microcomputing. Our number is (617) 256-5515.



CMC

# APPLE ANALOG INPUT

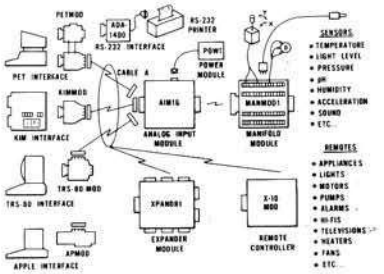
μMAC SYSTEMS

**Analog to Digital Conversion System for the APPLE Computer**

Give your APPLE computer the ability to measure and control the world around it with μMAC SYSTEMS modules. Just plug the APSET1 into your APPLE to get 16 channels of analog input. Screw terminals are provided for each channel so you can hook up pots, joysticks, thermometers, light probes, or whatever appropriate sensors you have.

Each of the 16 analog inputs, in the range of 0 to 5.12 volts, is converted to a number between 0 and 255 (20 millivolts per count).

Software is included.




APSET1

| | |
|---|---|
| 1-AIM16 | - 16 ANALOG INPUTS - 8 BITS - 100 MICROSEC |
| 1-APMOD | - APPLE TO μMAC INTERFACE |
| 1-CABLE A24 | - 24 INCH INTERCONNECT CABLE |
| 1-MANMOD1 | - MANIFOLD MODULE - SCREW TERMINALS FOR INPUTS, REFERENCE, GROUND |
| 1-POW1 | - POWER MODULE |

APSET1a for 110 VAC $295
APSET1e for 230 VAC $305

Order direct or contact your local computer store.

**CONNECTICUT microCOMPUTER, Inc.**
*150 POCONO ROAD*
*BROOKFIELD, CONNECTICUT 06804*
*TEL: (203) 775-9659 TWX: 710-456-0052*

VISA AND M/C ACCEPTED-SEND ACCOUNT NUMBER, EXPIRATION DATE AND SIGN ORDER.
ADD $3 PER ORDER FOR SHIPPING & HANDLING-FOREIGN ORDERS ADD 10% FOR AIR POSTAGE.

## OSI OS!

# SOFTWARE FOR OHIO SCIENTIFIC

Over 50 programs for C1, C2, C4 & Superboard, on tape and disk. All come with listings and compete documentation.

**GAMES - 4K - Tape**

CHESS FOR OSI - specify system $19.95

STARFIGHTER 5.95
Real time space war.

SEAWOLFE 5.95
Floating mines, three target ships, etc.

LUNAR LANDER 5.95
With full graphics

TEN TANK BLITZ 9.95
A sophisticated real time tank game.

**8K GAMES**

BACKGAMMON 9.95

BLACKJACK 6.95
Plays all Vegas rules

Add $1.00 each for Color/Sound

**UTILITIES**

C1P CURSOR CONTROL $9.95
gives real backspace, one key screen clear, and **midline editing**

RENUMBERER 5.95

SUPERUTILITY 12.95
Has Renumberer, Variable table maker and Search

**BUSINESS**

SMALL BUSINESS ANALYSIS 15.95
Does profit and loss, quick ratio, breakeven analysis and more. 13 pages of documentation.

STOCK PORTFOLIO 6.95
Keeps track of your investments

Our $1.00 catalog has free game and utility listings, programming hints and a lot of PEEKs and POKEs and other stuff that OSI forgot to mention - and a **lot** more programs for sale.

DISKS 5" COLOR/SOUND $29.95
DISK 1. STARFIGHTER, ROBOTANK, SEA WOLFE, BOMBER, TEN TANK BLITZ
DISK 2 BREAKTHROUGH, LUNAR LANDER, ALIEN INVADER, KILLERROBOTS, SLASHBALL

**AARDVARK**
**1690 Bolton, Walled Lake, Michigan 48088 • (313) 624-6316**



# DISCOUNT DATA PRODUCTS

## BASF 5¼" DISKETTES: $34.50 PER BOX OF 10

HIGHEST QUALITY DISKETTES AT A BARGAIN PRICE! LABELS AND WRITE-PROTECT TABS INCLUDED.

## VINYL DISKETTE HOLDERS FOR NOTEBOOKS

THE IDEAL WAY TO STORE DISKETTES. EACH VINYL PAGE HOLDS TWO DISKETTES AND INCLUDES A POCKET FOR EACH DISKETTE'S LABEL. SAFELY KEEP UP TO 40 DISKETTES IN A SINGLE 1" 3-RING NOTEBOOK!

## $4.95/SET OF 10

*MARKETING YOUR OWN SOFTWARE? DDP OFFERS DEALER & SOFTWARE HOUSE DISCOUNTS ON NOT ONLY THE ABOVE ITEMS, BUT ALSO THE FOLLOWING PRODUCTS:*

**9" x 12" ZIP-LOCK BAGS FOR PACKAGING & DISPLAY OF SOFTWARE.**

**CORRAGATED MAILERS TO SHIP TO USERS OR DEALERS!**

***SEND FOR FREE INFORMATION AT:***

**DISCOUNT DATA PRODUCTS**
P.O. BOX 19674-M
SAN DIEGO, 92119

(ADD $1.00 SHIPPING/HANDLING CHARGE TO ALL ORDERS.)



**6502 ENTHUSIASTS**

A NEW KIT THAT GETS YOU INTO SERIOUS MICROCOMPUTING FOR ONLY

$14.95 Kit $16.95 Assembled

> 6502 CPU & DISK CONTROLLER CARD
> 16K RAM CARD
> 90K MINIFLOPPY DISK DRIVE
> DOS, 8K BASIC, ASSEMBLER/EDITOR
> 8 SLOT MOTHERBOARD
> CHASSIS, POWER SUPPLY, & CABLES
> MICRO—TERM ACT—I SERIAL TERMINAL
16 X 64 CHARACTER DISPLAY
Upper/Lower CASE
ADDRESSABLE CURSOR
> GOLDSTAR TV/MONITOR
HIGH QUALITY DISPLAY
TV TUNER INTACT

Tek-Aids Industries Inc.

Master Charge
VISA
C.O.D.

**New Dimensions in Electronics**
44 University Drive • Arlington Heights, Ill. 60004 • (312) 870-7400



## OHIO SCIENTIFIC

**Hardware..**C1P Video—gives true 32 or 64 chrs/line with guard bands. This is not a make-shift mod. It makes your video every bit as good as the 4P's plus you have switch selectable 1,2 and 3 MHz. CPU clock as well as 300, 600 and 1200 baud for cassette and serial port all crystal controlled.

Complete plans—$18.95, Kit $13.95 or send in your C1P to Personal & Business Computer Connection, 38437 Grand River, Farmington Hills, Mich 48018, and we will install the Video mod for $79.95. Other mods available..add sound; RS-232 port cassette motor control.

**Software** (with documentation) For C1, C2, 4P & 8P Chess 1.9, Backgammon, excellent card games, arcade type games, utility programs, mini word processor memory maps, etc.

Catalog with free program (hard copy) & memory map for BASIC in ROM models...$1.00

**Progressive Computing**
**3336 Avondale Crt.**
**Windsor, Ontario**
**CANADA N9E 1X6**
(516)969-2500

# Shorthand Commands for Superboard II and Challenger C1P BASICs

**This article shows how to intercept the BASIC's input routine and how to implement a shorthand notation.**

**Henk J. Wevers**
**Cloeckendaal 38**
**6715 JH EDE**
**The Netherlands**

As a superboard or Challenger IP owner, you surely have noticed the large amount of adds for extra software for the Apple, PET and TRS 80 machines and you hoped for just some of these goodies to show up for your own computer.

Well, no such luck, so far. So, now we have to do the job ourselves. One of the advertised options for the TRS 80, single stroke instructions, looked nice and I started to program something like that for the OSI machine. The result is presented here, and the shorthand routine is almost alway present in my machine during program development.

Before describing how the job was done, let's first have a look at what this routine does exactly. After loading the program, type

```
POKE 536,34:POKE 537,2
```

and instead of typing the instruction letter by letter, you can enter it by hitting the ESCAPE KEY and another key after that. The last key determines which instructin is entered. For instance, if you want to enter RIGHT$, hit the ESCAPE key first. On the display the cursor will change to an arrow to warn you that the next entry will be an instruction instead of a single character. Now hit C and you have just entered RIGHT$ as the display shows you.

All instructions are accessible in this way, and by altering the table in memory locations 0280 through 02C3 you may even choose your own shorthand codes.

There are a few things about the Microsoft Basic for the OSI machines that should be known before you can fully understand the program.

First, if Basic asks for an input, the input routine is accessed by a vector located in memory locations 0218 and 0219 (hex) or 536 and 537 decimal. You can intercept the input by changing these locations and so routing the input through your own routines, and that is the way I did the job.

Secondly, to use the token system in their Basic, Microsoft put a table containing all possible instruction in their program starting at location A084 hex. The instructions are separated in the table by the last character of ever instruction having bit 7 set. If you strip off bit 7 of the token, you have the relative position of the instruction in the table. If we look at the instruction END with token 80, then this one has the first position in the table (actually position 0, since we count from zero). RIGHT$ with Token C3 (hex) has the hex position of 43 in the table.

Third to consider is that the input buffer is located at hex location 13 and up in page zero. X serves as the buffer pointer during input.

And lastly, location 0200 is used as the relative cursor location. The routine WRCHAR at BFC2 puts the character in location 0201 on the screen, at location D300 + the contents of 0200. You can use this in your own programs; I found it very handy. Now to the program. Most of it will be clear to you now.

First a character from the keyboard or cassetport is input, and if it is not 'ESCAPE' we return it to the A register. Basic can't tell the difference between this routine and the original one. If the character is 'ESCAPE', the cursor is changed from underline to right arrow and another character is input. The shorthand table (0280 through 02C3) is scanned for a match, and if a match is found, the X register will contain the token for that command with bit 7 stripped off. If no match is found another (shorthand) character is input. The start of the instruction is searched for in the Basic table and then the instruction is output to the screen and stored in the input buffer, character by character. If bit 7 of a character is set, (signaling the end of the instruction) this process is stopped, bit 7 stripped off and the last character processed. Another character or shorthand command can then be input.

By now you have noticed the strange BIT $07A9 instruction around locations 0263 and 0274. It is a short way of entering a routine with different contents of the accumulator. For instance, if you enter the OUTCH routine via locations 0262 - 0263 - 0266, you have the character in A output, but entering the routine via 0264-0266 you have the 'BELL' character in A and so output. 025E and 026F switch between the two, depending on the input buffer being full.

Now let's look at the shorthand table starting at location 0280. The last two characters of the addresses also give the token for the instruction. I have programmed this table for you in a way that I have found convenient for the location of the shorthand commands on the keyboard. If you want to program this layout yourself, just enter the ASCII value in the table for the shorthand key you want. For example, if you want the Q-key to be shorthand for 'THEN', only put 51, the ASCII code for Q, in location 02A0, the location for 'THEN'.

The last thing to explain is the choice of where to put this routine in memory. I used locations 0222 and up, because these locations are unused by BASIC and the monitor, and they are not affect by either a cold or a warm start. If you have hit the BREAK key you have to change the input vector again by proper POKING as described earlier.

I hope this little routine will make programming a little easier for you as it did for me. Imagine being able to RUN, LIST, SAVE, and LOAD with one simple keystroke! Most likely, you have exceeded the maximum line length by using a ? instead of PRINT, so you had to type the line all over again after a list, because the program wouldn't load. This routine shows PRINT on the screen after 'ESCAPE' and ? so you will always see what you are doing. Good luck!

**Second part of Memloc is taken for command in that location.**

| MEMLOC | COMMAND | SHORTHAND | CODE(HEX) |
|---|---|---|---|
| 0280 | END | H | 48 |
| 0281 | FOR | Q | 51 |
| 0282 | NEXT | G | 47 |
| 0283 | DATA | O | 4F |
| 0284 | INPUT | I | 49 |
| 0285 | DIM | J | 4A |
| 0286 | READ | U | 55 |
| 0287 | LET | ! | 21 |
| 0288 | GOTO | R | 52 |
| 0289 | RUN | 'CR' | OD |
| 028A | IF | D | 44 |
| 028C | GOSUB | T | 54 |
| 028D | RETURN | Y | 59 |
| 028E | REM | " | 22 |
| 028F | STOP | ^G | 07 |
| 0290 | ON | : | 3A |
| 0291 | NULL | ^E | 05 |
| 0292 | WAIT | ^A | 01 |
| 0293 | LOAD | L | 4C |
| 0294 | SAVE | K | 4B |
| 0295 | DEF | ^D | 04 |
| 0296 | POKE | A | 41 |
| 0297 | PRINT | ? | 3F |
| 0298 | CONT | ^B | 02 |
| 0299 | LIST | 'RUBOUT' | 7F |
| 029A | CLEAR | ^C | 03 |
| 029B | NEW | 'LF' | OA |
| 029C | TAB( | . | 2E |
| 029D | TO | W | 57 |
| 029E | FN | ^F | 06 |
| 029F | SPC( | ; | 3B |
| 02A0 | THEN | F | 46 |
| 02A1 | NOT | ) | 29 |
| 02A2 | STEP | E | 45 |
| 02A3 | + | + | 2B |
| 02A4 | - | - | 2D |
| 02A5 | * | * | 2A |
| 02A6 | / | / | 2F |
| 02A7 | ^ | ^ | 5E |
| 02A8 | AND | 5 | 35 |
| 02A9 | OR | % | 25 |
| 02AA | > | > | 3E |
| 02AB | = | = | 3D |
| 02AC | < | < | 3C |
| 02AD | SGN | ( | 28 |
| 02AE | INT | 6 | 36 |
| 02AF | ABS | & | 26 |
| 02B0 | USR | ' | 27 |
| 02B1 | FRE | 7 | 37 |
| 02B2 | POS | 8 | 38 |
| 02B3 | SQR | 9 | 39 |
| 02B4 | RND | Ø | 30 |
| 02B5 | LOG | $ | 24 |
| 02B6 | EXP | 4 | 34 |
| 02B7 | COS | 2 | 32 |
| 02B8 | SIN | 1 | 31 |
| 02B9 | TAN | 3 | 33 |
| 02BA | ATN | # | 23 |
| 02BB | PEEK | S | 53 |
| 02BC | LEN | M | 40 |
| 02BD | STR$ | B | 42 |
| 02BE | VAL | , | 2C |
| 02BF | ASC | N | 4E |
| 02C0 | CHR$ | V | 56 |
| 02C1 | LEFT$ | Z | 5A |
| 02C2 | RIGHT$ | C | 43 |
| 02C3 | MID$ | X | 58 |

SHORTHAND COMMAND FOR OSI CHALLENGER IP and SUPERBOARD

```
0222    20 BA FF  SHORT1   JSR  IN          INPUT CHAR FROM KEYB. OR TAPE
0225    C9 1B              CMPIM $1B         IS IT 'ESCAPE' ?
0227    F0 01              BEQ   SHORT2      YES? BRANCH
0229    60                 RTS               NO, RETURN TO BASIC WITH ECHAR
022A    98         SHORT2  TYA               SAVE Y
022B    48                 PHA               AND
022C    8A                 TXA               X REGISTERS
022D    48                 PHA
022E    A9 12              LDAIM $12         LOAD 'ARROW' TO
0230    8D 01 02           STA   CURSOR      CHANGE CURSOR
0233    20 C2 BF           JSR   WRCHAR      DO IT
0236    A2 43      SHORT3  LDXIM $43         LOAD MAX TABLE INDEX
0238    20 BA FF           JSR IN            INPUT SHORTHANDCOMMAND
023B    DD 80 02   SHORT4  CMPX  TABLE       COMPARE WITH TABLE
023E    F0 06              BEQ   SHORT5      FOUND IT? BRANCH
0240    CA                 DEX               DECREMENT INDEX FOR NEXT TRY
0241    10 F8              BPL   SHORT4      IF NOT DONE? LOOP BACK
0243    4C 36 02           JMP   SHORT3      NO MATCH? IGNORE AND LOOP BACK
0246    E8         SHORT5  INX               COME HERE WITH TABLE INDEX IN X
0247    A0 FF              LDYIM $FF         PREPARE FOR LOOKUP IN COMMAND T
0249    CA         SHORT6  DEX               COMMAND FOUND?
024A    F0 08              BEQ   SHORT8      YES? BRANCH
024C    C8         SHORT7  INY               NO SKIP CURRENT COMMAND IN TABL
024D    B9 84 A0           LDAY  $A084,Y     DONE YET?
0250    10 FA              BPL   SHORT7      NO, LOOP BACK
0252    30 F5              BMI   SHORT6      YES? GO AND TRY NEXT ITEM
0254    68         SHORT8  PLA               GET INPUTBUFFER INDEX BACK
0255    AA                 TAX               AND  STORE IT IN X REG
0256    C8         SHORT9  INY               GET READY TO STORE COMMAND IN B
0257    B9 84 A0           LDA $A084,Y       GET COMMAND CHAR
025A    30 0F              BMI   SHORT10     IF LAST CHAR OF COMM, BRANCH
025C    E0 47              CPXIM $47         INPUTBUFFER FULL?
025E    B0 04              BCS   +04         YES? BRANCH TO SHORT9A + 1
0260    95 13              STAX  $13         STORE CHAR IN INPUTBUFFER
0262    E8                 INX               INCR. BUFFERPOINTER
0263    2C A9 07   SHORT9A BIT   $07A9        SKIP OR LOAD 'BELL' IN A
0266    20 E5 A8           JSR   OUTCH       OUTPUT CHAR
0269    D0 EB              BNE   SHORT9      BRANCH ALWAYS
026B    29 7F      SHORT10 ANDIM $7F         LAST CHAR. STRIP OF HIGH BIT
026D    E0 47              CPXIM $47         INPUTBUFFER FULL?
026F    B0 04              BCS   + 04         YES, BRANCH
0271    95 13              STAX  $13         STORE CHAR IN INPUTBUFFER
0273    E8                 INX               INCR BUFFERPOINTER
0274    2C A9 07           BIT   $07A9       SKIP OR LOAD 'BELL' IN A
0277    20 E5 A8           JSR   OUTCH       OUTPUT CHAR
027A    68                 PLA               RESTORE
027B    A8                 TAY               Y REGISTER AND
027C    4C 22 02           JMP    SHORT1     LOOP BACK

0280 - 02C3        TABLE
```

```
POKE 536,34:POKE 537,2  PUTS SHORTHAND ON AND
POKE  536,186:POKE 537,255 OFF
```

# Classified Ads

OSI Superboard C1P Club International is oriented to your system. Join US, Canada, Europe & get giant manthly newsletter. All members MUST send at least 1 application, modification or Basic/Machine lang program + $12. US money order for yearly membrshp to:

S-BD C1P Club International
Bx 55, Agincourt, Ont.
CANADA MIS 3B4

APPLE GRAPHICS Tablet fill PGM Solid color fill of shapes. 1 pen press fills most shapes greater than 14 pixels wide. Reg. tablet, 48K, disk. $25 for pgm disk & instructions.

Dave McKee
10205 Rock Circle Dr.
Edmond, OK 73034

OSI: C1P8K $375, C4P8K $659, 610 board $288, C1P/C4P disk drive $479, 4K 2114 memory $58, other OSI available. Centronics 730 printer, 50 cps 80 col. tractor & platen feed only $875. Add shipping, personal checks take 3 wks to clear.

TRACK ZERO
1418 Hanson Drive
Normal IL 61761

MAJOR LEAGUE BASEBALL! Manage Major League baseball teams and make all decisions; including pitching, hitting, running and lineup changes. Includes data files on all 1979 teams and utility programs to create and edit your own team files. HIRES display. Apple II with 48K ROM Applesoft, Disk $25.

Stan Erwin
5410 W 20th St.
Indianapolis, IN 46224

OSI C1P Superboard II owners, you need the 96 page tutorial manual 'Getting Started with Your C1P'. Fundamentals of BASIC, cassette usage, subroutines, logic & control are described in step-by-step manner. $5.95 + $1 p&h form:

TIS
Box 921—M
Los Alamos, NM 87544

100 % PET disk oriented Macro Assmblr/Text Editor (MAE) Development software. Includes new version of ASSM/TED written for 32K new ROM PET & 2040 disk drive. Features macros, conditional and interactive assmbly, a relocating loader program. $169.95 for diskette and manual. Send $1.00 for details.

Eastern House Software
3239 Linda Drive
Winston-Salem, NC 27106

British APPLE Owners/Dealers! Write now to MGA for extensive list of specialized software and hardware for your APPLE or 2020. We promise you'll be surprised!

140 High Street
Tenterden, Great Britain
TN306HT

05—65D V3 Complete disassembled listing, fully commented, everything you always wanted to know but OSI wouldn't tell you. $19.95

Software Consultants
7053 Rose Trail
Memphis, TN 38134

# PERFECT AIM

**ATTRACTIVE FUNCTIONAL PACKAGING FOR YOUR AIM-65 MICROCOMPUTER**

- Professional Appearance
- Striking Grey and Black Color Combination
- Protects Vital Components

**ENGINEERED SPECIFICALLY FOR THE ROCKWELL AIM-65**

- All Switches Accessible
- Integral Reset Button Actuator
- Easy Paper Tape Replacement

**EASILY ASSEMBLED**

- Absolutely No Alteration of AIM-65 Required
- All Fasteners Provided
- Goes Together in Minutes

**MADE OF HIGH IMPACT STRENGTH THERMOFORMED PLASTIC**

- Kydex 100*
- Durable
- Molded-In Color
- Non-Conductive

**AVAILABLE FROM STOCK**

- Allow Three to Four Weeks for Processing and Delivery
- No COD's Please
- Dealer Inquiries Invited

TO ORDER: 1. Fill in this Coupon (Print or Type Please)
2. Attach Check or Money Order and Mail to:

NAME _____

STREET _____

CITY _____

STATE _____ ZIP _____

**SAE 1-1** PLEASE SHIP PREPAID _____ SAE 1-1(s)
@$43.50 each
California Residents Please Pay
$46.33 (Includes Sales Tax)

**SAE 1-2** PLEASE SHIP PREPAID _____ SAE 1-2(s)
@$46.50 each
California Residents Please Pay
$49.52 (Includes Sales Tax)

**enclosures group**

**771 bush street**
**san francisco, california 94108**

***TM Rohm & Hass** **Patent Applied For**

# A Formatted Dump Routine for the AIM 65

**This HEX dump utility permits the user to control the formatting of the dump to conform to his printer's capabilities.**

**W.E. Wilson**
**Washington State U.**
**Pullman, WA 99164**

The Dump routine in the AIM 65 Monitor produces a continuous character string and thus is not very readable. The dump format is essentially not fit for human consumption. The serious AIM 65 user who needs a memory dump is thus limited to using the Monitor "M" command, which only dumps four locations at a time. A more useful and efficient dump routine with a variable output format was needed by the author and thus the following program was written.

The Formatted Dump routine will dump memory over the range specified in response to the "FROM=" and "TO=" parameters. The number of bytes in each line of the dump is specified in response to "/". All input and output is in hexidecimal. Each line of the dump gives the starting address of the first byte in the line, a space, 1st byte, space, 2nd byte, etc. The standard AIM-65 printer will handle $05 bytes per line and an 80 column TTY type unit will handle up to $16 (22) bytes per line.

The dump routine makes extensive use of the routines in the AIM-65 Monitor as well as RAM locations reserved for the Monitor. No locations outside of the Monitor area, except for the dump routine itself, are used by the dump routine. Thus the dump routine may be located at any convenient place in RAM and will not affect any other software. The following dumps demonstrate the use of the routine.

AIM-65 MONITOR ROUTINES USED IN DUMP PROGRAM

E7A3 = Print "FROM=" and get address in $A41C/D.

E83E = Print " " (blank).

E910 = Move address from $A41C/D to $A41AB.

E7A7 = Print "TO=" and get address in A41C/D.

E837 = Print "/".

E785 = Get two hex digits and store in A419.

EA13 = Print "CRLF".

EA46 = Print one hex byte = Two ASCII characters.

EB58 = LDAY - Simulates LDA (N), Y without page 0.

E182 = AIM-65 Monitor Re-entry.

**A Formatted Dump Routine for the AIM-65**
**List 1**

```
0112 4C JMP 0F90



0F90 20 JSR E7A3
0F93 B0 BCS 0F90
0F95 20 JSR E83E
0F98 20 JSR F910
0F9B 20 JSR E7A7
0F9E B0 BCS 0F9B
0FA0 20 JSR E83E
0FA3 20 JSR E837
0FA6 20 JSR E785
0FA9 20 JSR EA13
0FAC AD LDA A41C
0FAF 38 SEC
0FB0 ED SBC A41A
0FB3 48 PHA
0FB4 AD LDA A41D
0FB7 ED SBC A41B
0FBA 30 BMI 0FF6
0FBC D0 BNE 0FC1
0FBE 68 PLA
0FBF F0 BEQ 0FF6
0FC1 AD LDA A41B
0FC4 20 JSR EA46
0FC7 AD LDA A41A
0FCA 20 JSR EA46
0FCD AE LDX A419
0FD0 A0 LDY #00
0FD2 20 JSR E83E
0FD5 A9 LDA #1A
0FD7 20 JSR EB58
0FDA 20 JSR EA46
0FDD C8 INY
0FDE CA DEX
0FDF D0 BNE 0FD2
0FE1 20 JSR EA13
0FE4 AD LDA A419
0FE7 18 CLC
0FE8 6D ADC A41A
0FEB 8D STA A41A
0FEE 90 BCC 0FF3
0FF0 EE INC A41B
0FF3 4C JMP 0FAC
0FF6 4C JMP E182
```

```
RUN
FORMATTED DUMP ROUTINE FOR THE AIM-65
ENTER VIA F3 FUNCTION KEY =↑
SPECIFY : FROM=,TO=,/=(CHRS/LINE)
     CHRS/LINE=TWO HEX DIGITS

<↑>FROM=B000  TO=B020  /05
B000  4C A3 CE 4C 7F
B005  B2 FE BE D1 C0
B00A  5D B6 5B B5 FF
B00F  BA 66 B7 BB B9
B014  D9 BD EF B9 13
B019  B8 13 B7 EB B6
B01E  96 B7 30 B6 F6

<↑>FROM=B000  TO=B020  /08
B000  4C A3 CE 4C 7F B2 FE BE
B008  D1 C0 5D B6 5B B5 FF BA
B010  66 B7 BB B9 D9 BD EF B9
B018  13 B8 13 B7 EB B6 96 B7

<↑>FROM=B000  TO=B020  /10
B000  4C A3 CE 4C 7F B2 FE BE D1 C0 5D B6 5B B5 FF BA
B010  66 B7 BB B9 D9 BD EF B9 13 B8 13 B7 EB B6 96 B7
```


# AIM 65 Software

NEW!

## * DISCOVER 6502 POWER *

**HELP!!**
9 Super utility programs for all AIM 65 programmers. **HEX INPUT:** Long and short versions, used for entering hex bytes into memory. **DUMP & HEXOUT:** Print out your memory in two formats for easy checking or location of individual bytes. **FIELD SORT:** A field sorting routine that finds usage in many tasks including helping you organize your programming. **RESTORE:** A program which automatically restores your editor after you've re-entered it improperly. This has been a real time saver for us. **ONE STEP:** Allows you to step thru the disassembly (K listing) one line at a time. **SYMBOL TABLE:** Is for use with the assembler ROM (How can you do without one?). It prints the beginning and ending addresses of your symbol table along with each label in your program and its address, all in a handy format. **RELOCATE:** Is a powerful program which allows you to move or relocate programs or data in memory. All who write, adapt or pirate programs or subroutines will appreciate this. It allows you to place them wherever you'd like. You can even open up spaces right in the middle of a program for inserting missing, new, or additional data or instructions. A programmers dream.

**GAIMS PAK I**
5 Exciting games of skill for 1 or several players, using the full capabilities of the AIM 65 keyboard, display, and printer. **HANGMAN:** A challenging word game for 2 players. The AIM does the work and keeps score. **SCORE 4:** A challenging game in 3 dimensions. The printer shows the positions of the 2 players after each move. **REACT:** Your reflexes are tested in thousandths of a second. The display and keyboard are turned into a reaction timer. **GOLLUMS CAVERNS:** Places you into the underground kingdom of the evil Wizard. You must move thru secret tunnels and cavern rooms avoiding traps, mysterious mist, and the Wizard's spell. To capture the Wizard you have only a few poison darts and your Magic computer to warn you when Evil is near. **BINHEX:** Teaches and tests your ability to convert Binary numbers into their Hexadecimal equivalents. Fun for budding programmers, and helps you to perfect a needed skill for fast and efficient programming.

**MATH WHIZ**
6 Programs dealing with numbers & math & the AIM 65. **ADD & SUBTRACT:** This powerful utility program turns your AIM 65 into a multiple precision calculator. **TOTAL:** Adds up to four decimal or hexadecimal numbers at a time. **TEST MEMORY:** Lets you really check out your RAM memory. **FIBBONACCI:** You learn about these important numbers as your AIM generates them in a series. **DEC TO HEX:** A multi-use program and algorithm for changing decimal numbers into their hex equivalents. **TIMER:** Makes your AIM 65 into a timer or a 12 or 24 hour clock, displaying or printing hours, minutes and seconds. A super demo of the power of the AIM 65.

YOU CAN NOW DEMONSTRATE THE POWER OF YOUR AIM 65 WITH CYBERDYNE'S DYNAMIC ACTION SOFTWARE. ALL OF OUR PROGRAMS RUN ON 1K OR 4K AIMS. ALL SOFTWARE ON EZ-LOAD TAPE CASSETTES. COMPLETE TEXT, PROGRAM, & LOADING INSTRUCTIONS ARE INCLUDED.

**"TAKE AIM" MANUAL, VOL I**
by JAMES HOYT CLARK of CYBERDYNE'S staff, coming soon—watch for it or write for info. A guide for all. Master AIM 65 hardware & software. A lab and learning manual, extension, clarification, & index to AIM 65 documentation. Over 30 programs. Explained & fully documented (games, math, utility, printer, display, & more).

**HOT LINE!!**

**(801) 224-2745**

**GAIMS PAK II**
6 Value packed games of skill and chance at less than $1.75 each. Created for maximum enjoyment of the AIM 65 by you and others. **BRICKS:** This program is unique because it learns from your mistakes and successes, while you play. It actually becomes "smarter" as you and the computer compete in a series of games. A real challenge to your skills of logic, deduction, and memory. **TIC-TAC-TOE:** Need we say more than the AIM 65 is a fair and impartial scorekeeper. **CARDS:** Gives you practice in when to hold-em and when to fold-em as your AIM 65 deals **5 CARD STUD** from an unmarked and randomly shuffled deck. **LOGICAL ORDER:** Tests your skills as a Master-Mind of reason and logic as you try to deduce a random 4 number sequence in the fewest number of tries. **STARWAY 090:** Places you at the controls of a crippled spacecraft. You must successfully pilot your craft back to the mothership for a soft rendezvous. Your supply of fuel is limited and must be used with care to avoid disaster. **ESP:** Even computers can have ESP (or seem to). You mentally pick a number, answer a few questions (without disclosing the number), and your AIM 65 will guess the number correctly every time.

**SHOW OFF**
7 Programs (less than $1.50 each) which show off all the features of the AIM 65. **SIGNS:** Lets you print 2 sizes of letters edgewise on the AIM 65 printer & make large banners. **ROTATING BILLBOARD:** Shows your messages as they rotate along the display. **PRINTER WAVE:** A good demo of user control of the AIM 65 printer. Starts you into graphics printing. **PAPER ADVANCE:** Gives you software control of the printer paper advance. See those last few lines printed for a change. **LINE FEED:** This time it is hardware control of the paper advance. **CURSOR DEMO:** You can light all 16 segments of the displays or type in different display patterns. **KEYBOARD INTERRUPT:** Gives you hardware control of the keyboard. Scans the keyboard for key closures without interrupting program execution. ALL PROGRAMS in this section may be used in your own programs as subroutines or run on their own as demos. Full instructions included.

**AIM 65 & 6502 RELATED PRODUCTS**
I/O-TTY-CASSETTE connector board for the AIM 65. Plugs directly to AIM 65 app connector (J1). Includes AIM connector, TTY and recorder jacks, cable set for 1 recorder, PC board with traces & holes for LED indicators, switch input sensors, optoisolators, drivers, relays, audio amp, & AIM 65 I/O training course with software. AIM I/O BOARD-Kit $19.75. Assembled $22.75.

**CYBERDYNES 1980 CATALOG.** HARDWARE, SOFTWARE, BOOKS, TRAINING ITEMS, GOODIES & R&D Services. (FULL DESCRIPTIONS) $1.00 (REFUNDABLE ON FIRST ORDER)—**FREE WITH ORDER.**

**ORDER NOW!**

- ORDER BY ITEM NAME AND PRICE. SOFTWARE CASSETTES $9.75 EACH. POSTAGE IN U.S.A. & CANADA 25¢ PER ITEM. OUTSIDE U.S. ADD 10% FOR AIR POSTAGE & HANDLING. FOR RUSH ORDERS PAY BY
- POSTAL MONEY ORDER. 10% DISCOUNT ON ORDERS OF 3 OR MORE ITEMS.

DEALER INQUIRIES INVITED. CUSTOM INDUSTRIAL MICRO-SYSTEMS OUR SPECIALTY.

## * SATISFACTION GUARANTEED *

Cyberdyne

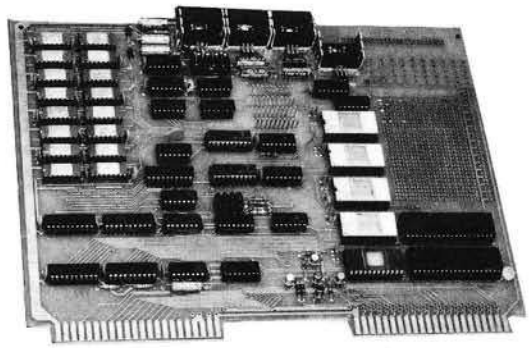
**P.O. Box 1285**
**Orem, Utah 84057**

# DRAM PLUS™

**DRAM PLUS [16K RAM]: TCB-101-16**
**DRAM PLUS [32K RAM]: TCB-101-32**

**DRAM PLUS offers the most powerful memory expansion capabilities available for the ASK family of microcomputers. Its many important features include:**

- **16K or 32K Dynamic RAM with all refresh handled on the board and completely transparent to the host microcomputer.**
- **Memory does not have to be addressed as a single 16K or 32K segment. 4K segments of memory may be placed on 4K boundaries.**
- **Up to 16K ROM/EPROM with provision for four ROMs or EPROMs: 2716/2516 2K EPROMs, 2732/2532 4K EPROMs, or 2332 ROMs. These may be mixed on the board.**
- **Two Versatile Interface Adapters, each with two 8-bit I/O ports, additional handshaking lines, two timers, and a serial-to-parallel shift register.**
- **Prototyping Area has space for adding circuits: memory write protection, floppy disk controller, communications devices, A/D or D/A, etc.**
- **EPROM Programmer handles all four types of EPROM: 2716/2516 2K and 2732/2532 4K.**
- **Simple Power Requirements of +5 volts at 1 amp and +12 volts at 150 milliamps. On board regulators permit unregulated power to be used.**



**Dynamic RAM Memory:** The RAM is composed of compact 4116 type dynamic memory. Each 4116 chip contains 16K bits, organized as 16K addresses with one bit of information per address. An 8 bit byte of memory is obtained by addressing 8 memory chips in parallel. Eight 4116 memory chips provide 16K bytes of memory. DRAM PLUS has provision for two sets of ram chips for a total RAM capacity of 32K bytes. All of the memory is socketted and each board is tested for the full 32K capacity. The only difference between the 32K and 16K versions is that 16K bytes of memory have been removed from the 16K version board. This memory may be added at any time.

**EPROM and/or ROM Memory:** There is provision for up to four EPROMs and/or ROMs to be added. These may be a mixture of the following types:

2516/2716 and 2532/2732 EPROMs or 2332 ROM.

The XX16 EPROMs contain 2K bytes and the XX32 contain 4K bytes. The 2332 is a 4K ROM. Using the 4K parts, up to 16K of read-only memory may be added.

**Versatile Interface Adaptor [VIA]:** Contains two 8-bit programmable I/O ports with additional handshaking lines; two timers; and a serial-to-parallel/parallel-to-serial shift register. The VIA may be used to interface keyboards, printers, and many other devices to the DRAM PLUS and/or the host system. A 24 page data sheet on the 6522 VIA is included in the documentation package.

**EPROM Programmer:** The VIA's are used in conjunction with some additional circuitry on the DRAM board to program all four types of EPROM. A separate programming socket is provided, a regulator circuit provides the programming voltage from a +27 volt input, and voltage to the EPROM is controlled by the program to prevent accidental damage to the EPROM.

**Transparent Refresh:** All of the circuitry for refreshing the dynamic RAM is contained on the board and operates in a manner that makes it completely transparent to the host microcomputer. All of the refresh memory accessing is done during Phase One, leaving the memory completely available to the host microcomputer during Phase Two. No "clock stretching" or "wait states" are required.

**RAM Memory Addressing:** Although the RAM is packaged as one or two 16K segments, provision has been made on DRAM PLUS for the memory to be addressed at four separately defined 4K boundaries per 16K segment. There are some restrictions on the set of boundaries that may be used within any 16K segment. Address bits A12 and A13 must not be the same for any of the 4K segments within a 16K segment. This results in a type of "Chinese Menu" selection. One 4K segment may be selected from each column of the following table, which lists the starting address of the 4K boundaries in hexidecimal:

| | | | |
|---|---|---|---|
| 0000 | 1000 | 2000 | 3000 |
| 4000 | 5000 | 6000 | 7000 |
| 8000 | 9000 | A000 | B000 |
| C000 | D000 | E000 | F000 |

An examination of the table will show that any four contiguous blocks will automatically come from different columns. If blocks were selected at 1000, 2000, and 3000, then the fourth block would have to be 0000 (which is highly unlikely on an AIM/SYM/KIM), 4000, 8000, or C000, for that 16K segment of memory.

**Prototype Area:** A prototyping area provides space and support for the addition of special circuits. The actual prototyping grid is approximately 2" by 2-3/4" and consists of a matrix of 13 by 28 holes spaced for standard sockets and IC's. The area is designed so that wirewrap or solder sockets may be used. The address and data lines are readily accessible to this area and convenient +5V and ground runs are provided. Connections to this area may be made through a separate connector facility which can support a standard connector with up to 50 pins.

**MICRO Bus Compatible:** The connections between the DRAM PLUS and the AIM/SYM/KIM follow the same conventions used by the original KIM-4 mother board. DRAM PLUS may be interfaced via a simple cable or the MOTHER PLUS.

**General Information:** The board is high quality, double sided with two sets of gold plated fingers with the same positioning as the connectors on the AIM/SYM/KIM. The board is the same size and shape as the SYM and KIM: 7-7/8" wide (excluding the edge connectors) by 10-3/4" long. All IC's are socketted to make field repair and servicing simple. Full documentation consists of instructions, schematics, program listings, data sheets and application notes. A Memory Test and an EPROM Programming Program are provided on a cassette tape which loads and works on the AIM/SYM/KIM. The DRAM PLUS Manual is available separately for $10.00, and this cost may be applied towards the purchase price.

**Trade-in:** You may be able to trade-in your Rev B **MEMORY PLUS** board toward a **DRAM PLUS**. Write/Call for details.

**DRAM PLUS with 16K RAM [TCB-101-16] $295.00 US/Canada, or $325.00 All Other Countries, plus shipping and handling.**
**DRAM PLUS with 32K RAM [TCB-101-32] $395.00 US/Canada, or $435.00 All Other Countries, plus shipping and handling.**
**Shipping/Handling: US $2.50 Canada $6.00 All Other Countries $18.00**

**This is one page from our eight page 1980 Catalog. Send name and address for your copy of the compete 1980 Catalog.**

**The COMPUTERIST, Inc., 34 Chelmsford Street, Chelmsford, MA 01824 617/256-3649**

## MICRO SUBSCRIBERS SPECIAL

### THANK YOU, COMMODORE ...

Just about a year ago, Skyles Electric Works introduced a 15 inch wide keyboard with all PET functions on nice, plump, full-sized keytops with torsion spring action. S.E.W. knew that most owners of the PET 2001-8 couldn't remain happy with the undersized keytops and unsatisfactory spring action.

Now you have announced that, effective in January of 1980, all PETs and CBMs would feature full sized keyboards.

So, thank you, Commodore, for confirming that S.E.W. was right all along.

Meanwhile, S.E.W. hasn't been resting...and now offers with the **Big KeyBoard** an 18 inch ribbon cable at no extra charge...and a cassette tape containing BASIC and machine language programs that convert the **Big KeyBoard** to ASCII or typewriter operation, including lower case characters, and upper case through the shift key. Even a shift lock, quotations in the proper place, numbers across the top row.

*All keytops double shot for lifetime durability. Switching action uses gold cross-point contacts; torsion springs are gold plated. Housing is black heavy gauge aluminum. Entire unit can be rinsed to clean under water and left to dry. 120 day warranty.* **Only $125.00***

**MICRO SUBSCRIBERS SPECIAL TILL MAY 15, 1980 $99.50***

# Wouldn't You Love Your PET 2001-8 Even More With the Skyles Big KeyBoard?

Enclosed is my MICRO address label showing subscription number, please send ___Big Keyboard(s).Enclosed is a check $_____*. Charge my VISA, Mastercharge (circle one) #_____ Expires_____

*California residents: please add 6% or 6.5% sales tax as required

**Skyles Electric Works** | 231E South Whisman Road
Mountain View, CA 94041
(408) 735-7891

# New and Better PET User Port Printer Routines

**A series of programs are presented which drive any TTL, parallel, or ASCII printer from the PET's user port.**

**Michael Tulloch**
**103 White Cr.**
**Niceville, FL 32578**

This article describes three programs which drive a printer from PET's user port. Any TTL, parallel, ASCII, printer can be driven. Two of the programs are in machine language and one is in BASIC.

Although there are several IEEE to serial and IEEE to parallel adaptors available for the Pet, the user's port is often needed to drive an ASCII device. In my case I saved $100 dollars by using the user port to drive my Trendcom printer. No hardware (except a cable and two connectors) is required. The software is equally simple: A printer driver with hand shake and a screen reader.

There are several reasons to drive your printer or other ASCII device from the user port. First, it is quick and easy. Second, some of the IEEE to ASCII adaptors respond to any and all device addresses. Third, if you already have an adaptor, the user port allows a tempory installation without interfering with existing devices. Another reason is that it allows you to have better and more direct control over the output. Both data and hand shaking can be done explicitly with software. Finally, and for me most importantly, it saves money. Just $2.19 for a ribbon cable and two junkbox connectors, had me printing

In general the following two programs comprise a screen printer. Two parameters can be adjusted by the calling program (or as direct commands): Start point, and ÷ of rows (if implemented in RAM), Thus a specific area, or window, of the screen can be printed. The two programs are named: 1. Printer Driver and 2. Screen Reader. For timing reasons Printer Driver is implemented only in machine language. Screen Reader, however, can be implemented either in machine language (Version A) or BASIC (Version B).

```
10000 POKE850,13:SYS849:
      FORR=0TO23:FORC=0TO39
10010 A=PEEK(32768+C+R*40)
10015 IF A=18 AND C=0 THEN STOP
10020 IF A<=31 THEN A=A+64::GOTO10060
10030 IF A<=63 THEN 10060
10040 IF A>127 THEN A=A-127:GOTO10060
10045 A=A+32
10060 POKE850,A:SYS826
10070 NEXT C:POKE850,13:SYS826
10080 NEXT R
10090 RETURN
READY.
```

***Figure 1***

Let's start with the easy one first— the BASIC Screen Reader. Figure 1 is a listing of this routine. Line 1000 clears the small printer buffer by making a carriage return and calling Printer Driver (located at 826 in this example). Line 10005 forms the screen reading loops with R the Row counter and C the Column counter. Here only eleven lines are printed. The Screen Value is placed into A by line 10010. Lines 10020 through 10045 convert the screen value to its equivelent ASCII code. Notice that graphic characters are printed as lowercase letters if they are on letter keys. Reversed letters are printed as not reversed letters, and not all graphic characters are printed. Figure 2 gives a sample of print out for the PET character set. The equivalant screen values are though 255. Version A is the machine language equivalent of Screen Reader. It's principle advantage is that it runs hundreds of times faster. In fact, on my Trendcome 100, which prints bidirectionally, you can't even see it hesitate between lines. At the Trendcom's 40 char/sec rate, a full screen of 1000 characters is printed in 30 seconds. Not Bad!

Another advantage is that you can hide it in the second cassette buffer and load it in only once. The BASIC version has to be attached to your program somehow.

A flow chart is shown in fiqure 3. It is annotated for the machine langage version. Figure 4 shows the dissassembled code. Figure 5 gives the HEX code as out put by the PET monitor program.

Block I initializes all registers. The screen read address is initialized to 32767. This is one less than the upper left screen start address value. Memory location 995 ($03E3) is the number of Rows to be printed. It's used as the Row counter. A column counter is stored in 992 (03E0). It is initalized with 40 and 40 is held in the X register for later use.

Block II increments the screen read address. Block III gets the screen value occupying the screen read address. This value is stored in location 996. Block IV is the adjustment routine. This is different from the scheme used in the BASIC program. Instead of using subtraction, addition is used. Although the logic is inverted from the BASIC program. the value

adjustments are the same. Critical temporary storage rigisters in addition to the program itself are listed in table I. The adjusted value is passed to the Printer Driver (Block V).

When control returns to Screen Reader the colums counter is decrimented. If the column counter is not equal to zero, then it is reset to the value stored in the X register (normally 40). Rows are then decrimented (Block IX)

Block X checks for the row counter equal to zero. If it is not, then a new screen value is read. If it is then that the program returns control to the calling routine.

The Printer Driver routine dissassembled listing is shown in figure 6. The PET monitor HEX dump is given in figure 7. Printer Driver takes a value (here stored in 85210), places it on the user port output lines, provides a data ready output pulse and waits for an ACKnowledge pulse. NOTE: If no ACK pulse is returned the program will continue to hold the PET off line. You must assure an ACK pulse will always be returned!

The above description of Printer Driver also sets up the Via registers. Each time it is entered it resets these registers for its own use. Only the E84C register is restored to its original value. Further, the routine inhibits interrupts. If an interrupt were to occur during the brief time a data ready pulse was being given, multiple outputs could be caused (and it does happen). There is also the chance that the ACK pulse would be missed, leaving PET in Limbo. Unfortunately this bit of protection has an adverse side affect. PET's internal clock will not keep correct time. Depending upon the amount of printing and your printer's characteristics this error can be substantial.

There are several improvements which could be made to these routines. Reversed character handling could be added to the "A" version of Screen Reader. Blanks could be output for nonprintable graphics characters. Codes could be developed for cursor command characters. You will probably want to make changes for your particular printer. There is room within Printer Driver to add a delay loop or NOPs to stretch the output pulse. Finally, Printer Driver can be used alone by passing ASCII values directly. Simply use PET's ASC ( ) command and Poke location 852.

Table I

| Decimal | Hex | Function |
|---|---|---|
| 1.2 | $1.2 | Screen read address |
| 992 | $03E0 | Column Counter |
| 993 | 03E1 | Row counter |
| 995 | 03E3 | Row input value |
| 996 | 03E4 | Screen value |
| 852 | 0354 | Value of output charater |

```
64@ 65A 66B 67C 68D 69E
70F 71G 72H 73I 74J 75K
76L 77M 78N 79O 80P 81Q
82R 83S 84T 85U 86V 87W
88X 89Y 90Z 91[ 92\ 93]
94^ 95_ 32  33! 34" 35#
36$ 37% 38& 39' 40( 41)
42* 43+ 44, 45- 46. 47/
480 491 502 513 524 535
546 557 568 579 58: 59;
60< 61= 62> 63? 96` 97a
98b 99c 100d 101e 102f 103g
104h 105i 106j 107k 108l 109m
110n 111o 112p 113q 114r 115s
116t 117u 118v 119w 120x 121y
122z 123{ 124| 125} 126~ 127
128 129 130 131 132 133
134 135 136 137 138 139
140 141
142 143 144 145
146 147 148 149 150 151
152 153 154 155 156 157
158 159 65A 66B 67C 68D
69E 70F 71G 72H 73I 74J
75K 76L 77M 78N 79O 80P
81Q 82R 83S 84T 85U 86V
87W 88X 89Y 90Z 91[ 92\
93] 94^ 95_ 32  33! 34"
```

```
35# 36$ 37% 38& 39' 40(
41) 42* 43+ 44, 45- 46.
47/ 480 491 502 513 524
535 546 557 568 579 58:
59; 60< 61= 62> 63? 96`
97a 98b 99c 100d 101e 102f
103g 104h 105i 106j 107k 108l
109m 110n 111o 112p 113q 114r
115s 116t 117u 118v 119w 120x
121y 122z 123{ 124| 125} 126~
127 128 129 130 131 132
133 134 135 136 137 138
139 140 141
142 143 144
145 146 147 148 149 150
151 152 153 154 155 156
157 158 159 65A 32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32  32  32  32  32
32  32
```

*Figure 2*

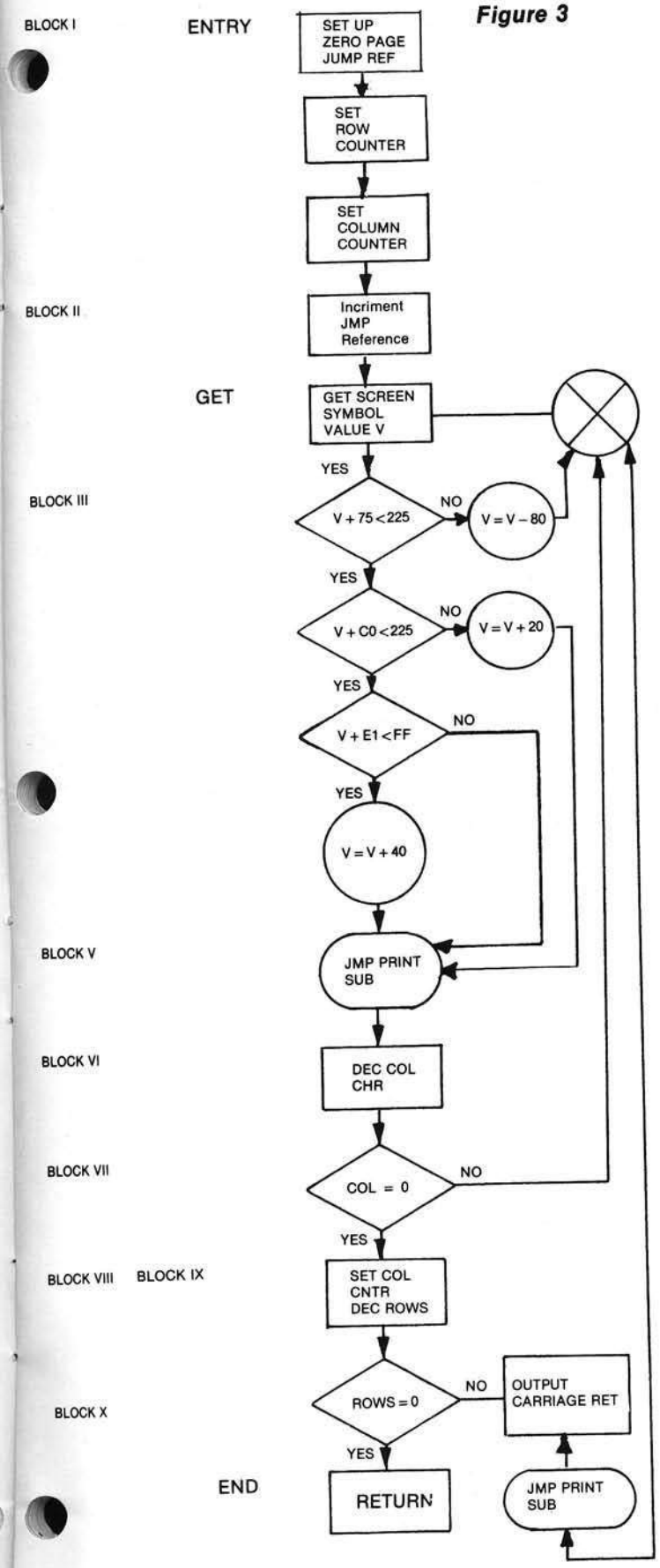



***Figure 3***

```
                                SCREEN READER

START  2884B            END     28985

TIME 160228             DATE 091779
```

***Figure 4:*** **Machine Language Listing of Version A Screen Reader. Note that the listing is for a high memory location. Addresses found at the following hex lines must be changed to relocate the program: $70f9, $70fc, $7108, $7113, $7129, $7134, $7137.**

```
2884A 704E  A0 00    LDYIM   00     0
28848 7030  A2 FF    LDXIM   FF     255
28850 70B2  8E 01 00STX      0001   1
28853 70B5  A2 7F    LDXIM   7F     127
28855 70B7  8E 02 00STX      0002   2
28858 70BA  AE E3 03LDX      03E3   995
28861 70BD  8E E1 03STX      03E1   993
28864 70C0  A2 28    LDXIM   28     40
28866 70C2  EA       NOP
28867 70C3  8E E0 03STX      03E0   992

28870 70C6  18       CLC
28871 70C7  A0 01 00LDA      0001   1
28874 70CA  69 01    ADCIM   01     1
28875 70CC  8D 01 00STA      0001   1
28878 70CF  A0 02 00LDA      0002   2
28882 70D2  69 00    ADCIM   00     0
28884 70D4  8D 02 00STA      0002   2
28887 70D7  18       CLC
28888 70D8  B1 01    LDAIY   01     1
28890 70DA  8D E4 03STA      03E4   996

28893 70DD  69 7F    ADCIM   7F     127
28895 70DF  B0 1F    BCS     1F     31
28897 70E1  A0 E4 03LDA      03E4   996
28900 70E4  69 C0    ADCIM   C0     192
28902 70E6  B0 25    BCS     25     37
28904 70E8  A0 E4 03LDA      03E4   996
28907 70EB  69 E1    ADCIM   E1     225
28909 70ED  B0 37    BCS     37     55
28911 70EF  A0 E4 03LDA      03E4   996
28914 70F2  69 40    ADCIM   40     64

28916 70F4  EA       NOP
28917 70F5  EA       NOP
28918 70F6  8D 54 03STA      0354   852
28921 70F9  20 3A 71JSR      713A   28986
28924 70FC  4C 18 71JMP      7118   28952
28927 70FF  EA       NOP
28928 7100  EA       NOP
28929 7101  A0 E4 03LDA      03E4   996
28932 7104  38       SEC
28933 7105  E9 80    SBCIM   80     128

28935 7107  18       CLC
28936 7108  4C 09 70JMP      7009   28937
```

```
28939 716B  EA       NOP
28940 716C  EA       NOP
28941 716D  AD E4 03LDA   03E4   996
28944 7110  18       CLC
28945 7111  69 20    ADCIM  20     32
28947 7113  4C F6 70JMP   70F6   28918
28950 7116  EA       NOP
28951 7117  EA       NOP

28952 7118  CE E0 03DEC   03E0   992
28955 711B  D0 AA    BNE    AA     170
28957 711D  8E E0 03STX   03E0   992
28960 7120  CE E1 03DEC   03E1   993
28963 7123  D0 09    BNE    09     9
28965 7125  60       RTS
28966 7126  AD E4 03LDA   03E4   996
28969 7129  4C F6 70JMP   70F6   28918
28972 712C  EA       NOP
28973 712D  EA       NOP

28974 712E  EA       NOP
28975 712F  A9 0D    LDAIM  0D     13
28977 7131  8D 54 03STA   0354   852
28980 7134  20 39 71JSR   7139   28985
28983 7137  4C C6 70JMP   70C6   28870
```

***Figure 5***

```
READY.

US S PC  SR AC XR YR SP
.; ;0EU 30 38 7E 31 FE
.  M 70AE,7137
            0  1  2  3  4  5  6  7
.:  70AE A0 00 A2 FF 8E 01 00 A2
.:  70B5 7F 8E 02 00 AE E3 03 8E
.:  70BE E1 03 A2 28 EA 8E E0 03
.:  70C6 18 AD 01 00 69 01 8D 01
.:  70CE 00 AD 02 00 69 00 8D 02
.:  70D6 00 18 B1 01 8D E4 03 69
.:  70DE 7F B0 1F AD E4 03 69 C0
.:  70E6 B0 25 AD E4 03 69 E1 B0
.:  70EE 37 AD E4 03 69 40 F4 EA
.:  70F6 8D 54 03 20 3A 71 4C
.:  70FE 71 EA EA AD E4 03 38
.:  7106 80 18 4C DA 70 EA EA AD
.:  710E E4 03 18 69 20 4C F6 70
.:  7116 EA EA CE E0 03 D0 AA 8E
.:  711E E0 03 CE E1 03 D0 09 60
.:  7126 AD E4 03 4C F6 70 EA EA
.:  712E EA A9 0D 8D 54 03 20 3A
.:  7136 71 4C C6 70 A9 FF 8D 43
.  X
BS
```

```
28985 7143  A9 FE    LDAIM  FE     254

                PRINTER DRIVER

START  28986          END     29051

TIME 160614           DATE 09/17/79
```

***Figure 6***

```
28986 713A  A9 FF    LDAIM  FF     255
28988 713C  8D 43 E8STA   E843   59459
28991 713F  AD 4C E8LDA   E84C   59468
28994 7142  48       PHA
28995 7143  A9 FE    LDAIM  FE     254
28997 7145  8D 4C E8STA   E84C   59468
29000 7148  AD 4B E8LDA   E84B   59467
29003 714B  29 E3    ANDIM  E3     227
29005 714D  8D 4B E8STA   E84B   59467
29008 7150  EA       NOP

29009 7151  EA       NOP
29010 7152  78       SEI
29011 7153  AD 54 03LDA   0354   852
29014 7156  8D 41 E8STA   E841   59457
29017 7159  AD 4C E8LDA   E84C   59468
29020 715C  29 1F    ANDIM  1F     31
29022 715E  09 C0    ORAIM  C0     192
29024 7160  8D 4C E8STA   E84C   59468
29027 7163  EA       NOP
29028 7164  EA       NOP

29029 7165  EA       NOP
29030 7166  EA       NOP
29031 7167  AD 4C E8LDA   E84C   59468
29034 716A  09 E0    ORAIM  E0     224
29036 716C  8D 4C E8STA   E84C   59468
29039 716F  EA       NOP
29040 7170  18       CLC
29041 7171  EA       NOP
29042 7172  AD 4D E8LDA   E84D   59469
29045 7175  29 02    ANDIM  02     2

29047 7177  F0 F9    BEQ    F9     249
29049 7179  68       PLA
29050 717A  58       CLI
29051 717B  60       RTS
29052 717C  EA       NOP
29053 717D  EA       NOP
29054 717E  EA       NOP
29055 717F  00       BRK
29056 7180  00       BRK
```

```
READY.
RUN

   PC  SR AC XR YR SP
.; C5ED 30 38 7E 31 FE
.  M 713A,7180
            0  1  2  3  4  5  6  7
FSUSD MONITOR
LOADING
READY.
SYS

   PC  SR AC XR YR SP
.; C5ED 30 38 7E 31 FE
.  M 713A,7189
            0  1  2  3  4  5  6  7
.: 713A A9 FF 8D 43 E8 AD 4C E8
.: 7142 48 A9 FE 8D 4C E8 AD 4B
.: 714A E8 29 E3 8D 4B E8 EA EA
.: 7152 78 AD 54 03 8D 41 E8 AD
.: 715A 4C E8 29 1F 09 C0 8D 4C
.: 7162 E8 EA EA EA EA AD 4C E8
.: 716A 09 E0 8D 4C E8 EA 18 EA
.: 7172 AD 4D E8 29 02 F0 F9 68
.: 717A 58 60 EA EA EA 00 00 24
.  X
READY.
>S
```

***Figure 7***


## General Purpose IO Board
for
## APPLE II™

* 2 8 bit programmable IO ports
* 1 timer/square wave generator
* 1 timer/counter
* 1 shift register
* 2 IO cables 50 pages instructions
* Large area for user buffers, relays, etc.

Board uses 6522 VIA See MICRO, 13:41, 15:17, 17:27, 1979

Order AP 1.0 $69.50

## Extender Card for APPLE II™

Lifts IO boards 4" above chassis

All lines labeled and numbered

Convenient test points for all lines

Order AP 2.0 $24.50

Orders post paid in US

Texas residents add 5% tax

MC-VISA orders must give all card data

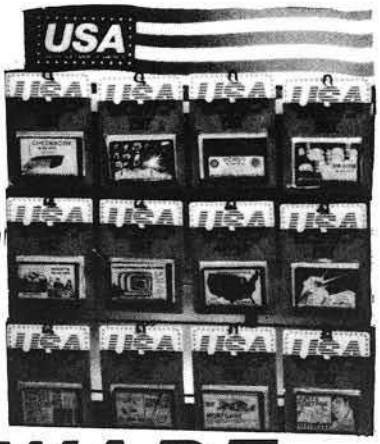
microAustin

PO Box 14408

Austin, Texas

78761






# GREAT PET SOFTWARE

"Precise, humanized, well documented an excellent value" are the applauds now being given to United Software's line of software. These are sophisticated programs designed to meet the most stringent needs of individuals and business professionals. Every package is fully documented and includes easy to understand operator instructions.

**DATABASE MANAGEMENT SYSTEM** - A comprehensive, interactive system like those run on mainframes! Six modules comprising 42K of programming allow you to; create, edit, delete, display, print, sort, merge, etc., etc. - databases of up to 10,000 records. Printer routines automatically generate reports and labels on demand. **60** pages of concise documentation are included. Requirements - 16-32K PET and 2040 Dual Disk (printer optional). . . .**Cost $125**

**ACCOUNTS RECEIVABLE/PAYABLE** - A complete, yet simple to use accounting system designed with the small businessman in mind. The United Software system generates and tracks purchase orders and invoices all the way through posting "controlled" accounts payable and accounts receivable subsystems.
Keyed Random Access file methods makes data access almost instantaneous. The low-cost solution for the first time computer user with up to 500 active accounts. Requirements - 32K PET, Dual Disk, any 80-column printer. . . .**Cost $175**

**CASH RECEIPTS & DISBURSEMENTS** - Makes it a breeze to track all outgoing payments made by any type of business operation. Checks are tracked by number and categorized by type of expense. Sorting, summary, and audit trails make it easy to post to general ledger. This system also categorizes incoming receipts. Uses KRAM file access method. Requirements - 32K PET, Dual Disk (printer optional). . . .**Cost $99.95**

**KRAM - Keyed Random Access Method** - The new, ultra-fast access method for the PET Disk, provides keyed retrieval/storage of data, in either direct or sequential mode, by either full or partial key values. Written by United Software in 6502 machine code, and designed with the PET in mind, it exploits all the benefits of the PET Disk, allowing full optimization of your system. Eliminates the need for "Sort" routines! KRAM provides flexibility never seen on a micro before. KRAM is modeled after a very powerful access method used on large-scale IBM Virtual Storage mainframes. So "KRAM" all you can into your PET - it will love you for it. . . .**Cost $79.95**

(Sublicenses available to software houses.)

**PROGRAMS FOR ENTERTAINMENT**

| | |
|---|---|
| Space Intruders ("Best Game of 1979") | $19.95 |
| Jury/Hostage | 12.50 |
| Kentucky Derby/Roulette | 9.95 |
| Alien I.Q./Tank | 9.95 |
| Tunnelvision/Maze Chase | 14.95 |
| Submarine Attack | 9.95 |
| Battle of Midway | 7.95 |
| Laser Tank Battle | 9.95 |
| Swarm | 14.95 |
| Super Startrek | 14.95 |
| PET Music Box | 29.95 |

**UNITED SOFTWARE PROGRAMS FOR BUSINESS**

| | |
|---|---|
| Checkbook | $15.95 |
| Mortgage | 15.95 |
| Finance | 12.95 |
| Bonds | 12.95 |
| Stock Analyzer | 22.95 |
| Stock Options | 24.95 |
| 6502 Macro Assembler | 49.95 |

Look for the RED-WHITE-BLUE United Software Display at your local computer dealer, or send check or moneyorder, plus $1.00 shipping to:

**UNITED SOFTWARE OF AMERICA**
750 Third Ave.
New York, N.Y. 10017

Dealer inquiries invited

# DATA CAPTURE 3.0

Is DATA CAPTURE just another smart terminal program for your Apple II™ or Apple II Plus™?

NO. It is a GENIUS TERMINAL PROGRAM and is designed to be used with the Micromodem II™.

Tired of watching data and programs scroll off the screen forever? Then DATA CAPTURE is the program for you.

- ANYTHING that appears on the screen of your Apple II can be captured. Any program or data.
- You can then save what you have captured to disk, dump it to your printer or even do simple editing with DATA CAPTURE.
- You can use DATA CAPTURE to compose text offline for later transmission to another computer. Think of the timeshare charges this will save you.
- Use DATA CAPTURE with the Dan Paymar Lower Case Adapter and you can enter lower case from the keyboard for transmission to another computer or capture both upper and lower case.
- A program is also included to convert your programs to text files for transmission using DATA CAPTURE.
- You receive two versions of the program. One is for upper case only and one for both upper and lower case use with the above adapter.

DATA CAPTURE will save you money if you are using a timesharing system because you can compose messages offline for later transmission. You can also quickly capture data for later reading, printing or editing. Requires DISK II™, APPLESOFT II™.

Price $29.95

If your local dealer does not have DATA CAPTURE then order directly. We ship DATA CAPTURE within 3 working days of receipt of order and welcome your personal check. We also accept Visa and Master Charge. Add $49.95 if you would also like to order the Dan Paymar Lower Case Adapter at the same time.

Ask for a catalog of our software.

* Apple II, Apple II Plus, Disk II and APPLESOFT II are trademarks of Apple Computer Company.

* Micromodem II is a trademark of D.C. Hayes Associates, Inc.

SOUTHEASTERN SOFTWARE
7270 Culpepper Drive
New Orleans, LA 70126

504/246-8438 504/246-7937

### Symbol-Table Sorter/Printer for the AIM Assembler

by Mel Evans
[MICRO 20:43]

"After more extended use of the program,I have found the following pair of bugs.

The first can be cured by replacing the code shown in Figure 1A (occurring at the end of subroutine SORT) by that in Figure 1B. The old code works often, but not always. The new code always works.

The second bug won't show until you start getting fancy with your source code. I was mistaken in thinking that memory locations 003C, 003D contain the address of the last symbol found during assembly. Instead, they contain the address of the last *active* symbol. With straightforward code, these will be one and the same. But suppose you have written your last subroutine (let's call it SUBZ) and then decide to initialize a couple of zero-page addresses (starting at ZP1) as in Figure 2A. After assembly, the last symbol will be SUBZ, but the last *active* symbol with be ZP1. And with this stored in 003C, 003D, you will get a very short listing!

The problem could be solved by re-writing the program to avoid using 003C, 003D. But, ther's a simpler solution, as shown in Figure 2B. Add a new symbol, LAST, as the last byte of the program. (This is a good practice anyway. After assembly, the address of LAST tells you precisely how much memory the program needs.) The, after initialization and any other housekeeping, add the line "* = LAST". This makes "last active" equal "LAST", and the listing comes out complete."

Submitted by: Mel Evans
ERIM, P.O. Box 8618
Ann Arbor, MI 48107

```
JSR CRCK
JSR INCADR
BMI PRNT1
BEQ PRNT1
JSR GAP
RTS
```

Figure 1A: Old SORT Code

```
JSR CRCK
TXA
BNE FIN
JSR INCADR
BNE PRNT1
DEX
BNE PRNT1
FIN JSR GAP
RTS
```

Figure 1B: New SORT Code

```
SUBZ
  .
  .
RTS
:
*=ZP1
.DBY $0A0B
;
.END
```

Figure 2A: Wrong "Last Active"

```
SUBZ
  .
  .
LAST RTS
;
*=ZP1
.DBY $0A0B
;
*=LAST
;
.END
```

Figure 2B: Right "Last Active"

# Microbes and Updates

### Expanding the SYM - 1 ... Adding an ASCII Keyboard

by Robert A. Peck
[MICRO 21:5]

"As we discussed, here is a corrected version of my progrm listing. Somehow the hex locations column of this listing was not used for the article. [Sorry about that - MICRO] Typos corrected on final version including label "DISP" change to WAIT2 at location 206 (minor), incorrect object code fixed at line 222 to **20 47 8A** .... Last was pointer to KSTAT at line 240 which should be **39**."

Submitted by: Robert A. Peck
P.O. Box 2231
Sunnyvale, CA 94087

```
0200  20 88 81  GKEY   JSR  SAVER   SAVE REGISTERS
0203  AD 01 A8         LDA  A801    GET PARALLEL ASCII
0206  F0 24            BEQ  WAIT2   UNLESS NONE,THEN BRANCH
0208  85 F1            STA  00F1    STORE IT A WHILE
020A  A9 10            LDA  #$10    DEBOUNCE CONSTANT
020C  85 EF            STA  00EF    DEBOUNCE
020E  C6 F0      WAIT1 DEC  00F0    SMALL LOOP
0210  D0 FC            BNE  WAIT1
0212  C6 EF            DEC  00EF    LARGE LOOP
0214  D0 F8            BNE  WAIT1
0216  20 03 89  SCANA  JSR  IJSCNV  SCAN DISPLAY(USE SCANVEC)
0219  2C 01 A8         BIT  A801    IS KEY STILL DOWN?
021C  30 F8            BMI  SCANA   WAIT FOR KEY RELEASE
021E  A5 F1            LDA  00F1    KEY UP, PROCESS KEY
0220  29 7F            AND  #$7F    STRIP KEY STROBE BIT
0222  20 47 8A         JSR  OUTCHR  SEND INTO DISBUF
0225  A5 F1            LDA  00F1    GET IT AGAIN
0227  29 7F            AND  #$7F    STRIP IT AGAIN
022A  4C B8 81         JMP  RESXAF  RETURN WITH ASCII IN A
022C  A9 10      WAIT2 LDA  #$10    IF NO KEY,
022E  85 EF            STA  00EF    SCAN DISPLAY
0230  20 03 89  SCANB  JSR  IJSCNV  THRU SCANVEC
0233  C6 EF            DEC  00EF    A NUMBER OF TIMES
0235  D0 F9            BNE  SCANB   THEN GO BACK
0237  F0 CA            BEQ  GKEY    AND LOOK AGAIN
0239  AD 01 A8  KSTAT  LDA  A801    READ ASCII INPORT
023C  0A               ASLA         SHIFT MSB INTO CARRY
023D  60               RTS          RET, CFLAG=1 IF KEY DN.

0240  20 86 8B  INIT   JSR  ACCESS  UNPROTECT SYSRAM
0243  A9 00            LDA  #00     MODIFY
0245  8D 61 A6         STA  A661    KEYBOARD
0248  A9 02            LDA  #02     INPUT
024A  8D 62 A6         STA  A662    VECTOR
024D  A9 39            LDA  #$39
024F  8D 67 A6         STA  A667    KEYPRESS
0252  A9 02            LDA  #02     STATUS
0254  8D 68 A6         STA  A668    VECTOR
0257  4C 03 80         JMP  WARM    WARM ENTRY,MONITOR
```

**FREE !** up to $170. in merchandise with purchase of PET-CBM item ! ! !

FREE MERCH

| | | |
|---|---|---|
| PET 16K Large Keyboard | $ 995 | $130 |
| PET 32K Large Keyboard | $1295 | $170 |
| PET 8K Large Keyboard (New) | $ 795 | $100 |
| PET 2040 Dual Disk (343K) | $1295 | $170 |
| PET 2023 Printer (pres feed) | $ 695 | $ 70 |
| PET 2022 Printer (trac feed) | $ 795 | $100 |

KIM-1 $159 (Add $30 for Power Supply) SYM-1 $ 209.00
AXIOM EX-801 Printer-PET ..... $ 477.00
2114 L 450 ns ..... 5.35 24/4.95 100/4.45
2716 EPROM (5 Volt) ..... 29.00
6550 RAM (for 8K Pet) ..... 12.70
PET 4 Voice Music System (KL-4M) ..... 34.50
All Books and Software ..... 15% OFF
Leedex Video 100 12" Monitor ..... 119.00

**ATARI — INTRODUCTORY SPECIAL**

ATARI 400, Atari 800, and all Atari Modules 20% OFF.

Heath WH-19 Terminal (fact. asm.) ..... 770.00
Programmers Toolkit - PET ROM Utilities ..... 44.90
PET Word Processor - Machine Language ..... 24.00

SALE

3M "Scotch" 8" Disks ..... 10/31.00
3M "Scotch" 5" Disks ..... 10/31.50
Verbatim 5" Disks ..... 10/26.50
Disk Storage Pages ..... 10/ 3.95

Cassettes (all tapes guaranteed) Premium quality, high output low noise in 5 screw housing with labels: AGFA PE 611

| | | | |
|---|---|---|---|
| C-10 | 10/5.95 | 50/25.00 | 100/48.00 |
| C-30 | 10/7.00 | 50/30.00 | 100/57.00 |

*Add $1 per order for UPS shipping.*
Ask for 6502, TRS-80, and S-100 Product List.

**A B Computers** 115 E. Stump Road, Montgomeryville, PA 18936, (215) 699-8386



**DISK DRIVE WOES? PRINTER INTERACTION?**
**MEMORY LOSS? ERRATIC OPERATION?**
**DON'T BLAME THE SOFTWARE!**

ISO-1

ISO-2

Power Line Spikes, Surges & Hash could be the culprit!
Floppies, printers, memory & processor often interact!
Our unique ISOLATORS eliminate equipment interaction AND curb damaging Power Line Spikes, Surges and Hash.

*ISOLATOR (ISO-1A) 3 filter isolated 3-prong sockets; integral Surge/Spike Suppression; 1875 W Maximum load, 1 KW load any socket ..... $56.95
*ISOLATOR (ISO-2) 2 filter isolated 3-prong socket banks; (6 sockets total); integral Spike/Surge Suppression; 1875 W Max load, 1 KW either bank ..... $56.95
*SUPER ISOLATOR (ISO-3), similar to ISO-1A except double filtering & Suppression .... $85.95
*ISOLATOR (ISO-4), similar to ISO-1A except unit has 6 individually filtered sockets .... $96.95
*ISOLATOR (ISO-5), similar to ISO-2 except unit has 3 socket banks, 9 sockets total ... $79.95
*CIRCUIT BREAKER, any model (add-CB) Add $ 7.00
*CKT BRKR/SWITCH/PILOT any model (-CBS) ..... Add $14.00

PHONE ORDERS 1-617-655-1532

**Electronic Specialists, Inc.**
171 South Main Street, Natick, Mass. 01760

Dept. MI



**BACLAN** would like to know if you

**WANT TO PROCESS DATA ON YOUR APPLE?**

- if so you should be looking for efficient tools to assist with data entry, (i.e. building files) and file handling (i.e. scanning, sorting, printing and copying files).

**and**

- if you are also looking for economy, we think you will be pleasantly surprised by the low price of the

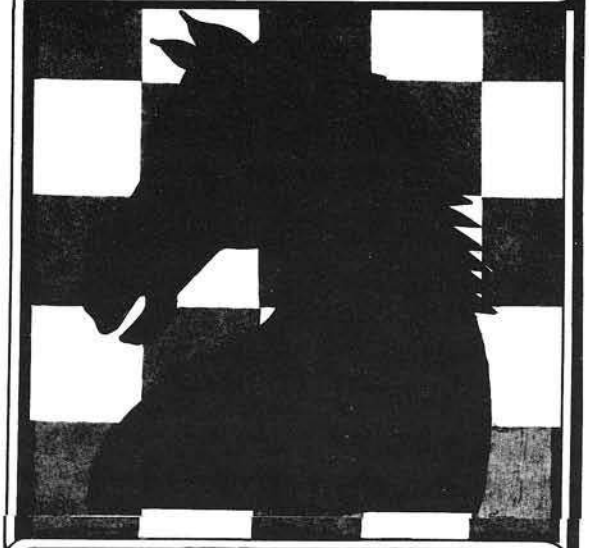
**BACLAN FILE HELPER**

available at your Apple Computer Dealer in both Applesoft and Integer Basic versions

**BACLAN** P.O. Box 36
(301) 997-9610 Columbia, MD. 21045



**MICROCHESS**
**for the AIM**



**MICRO SOFTWARE**

**AIM MicroChess with Player's and Programmer's Manual, complete Source Listings, Object on Cassette Tape. $15.00 plus shipping [$1.00 US/$2.00 Anywhere Else]**
**MICRO Software, P.O. Box 6502, Chelmsford, MA 01824**

# Graphics and the Challenger C1P, Part 5

**This final installment in the series discusses plotting techniques and moving characters.**

William L. Taylor
246 Flora Road
Leavittsburg, OH 44430

The ability to have characters and have them move in our game programs is a must. How do we accomplish this task? It is a simple task to implement. We do it with a technique called plotting.

### C1P Plotting Technique

In order to have any character move on our C1P's Monitor screen, we must first know where we wish our character to start, the angular directions in which it is to move, and where the character's movement will end. If you will examine the example of the plotting diagram in Figure 1, you can see the angular directions in which the character can be made to move on the monitor screen. These angular directions are relative to any point on the screen, i.e., relative to a certain position on the Video RAM. If, for example, the starting location were 54000 decimal, the zero point would be 54000 decimal and all movement would be relative to that point.

As stated in an earlier part of this series, we can cause a character to be placed on the screen of our C1P with a BASIC POKE statement. We move the character that has been placed on the screen with a BASIC FOR/NEXT loop. In the explanation of plotting and how to develope animated characters we will use the functions of BASIC to develope our programs and to describe animation methods.

To begin our explanation, let's use decimal location 54000 as an example again as a starting point. A BASIC program would use this decimal location as a variable content. For example, 10 A = 54000. Now that we have a starting point, we can move the graphics character in any direction shown in the diagram in Figure 1. For example, if you wish the character to move in a vertical direction, with a BASIC subroutine we can get the character displayed and moved. In order to explain how this proceedure works, please refer to the BASIC program subroutines in Listing 1, along with Figure 1.

If we wish to have an animated character (one that moves) we must first know the start, end, and path of the character, as stated before. The character must be made to appear at a point along the path of angular movement. The character is then displayed for some duration of time. Next it is erased from its present position and then displayed at a new position on the monitor screen. This process must be continued for the desired distance along the plotted path that we have chosen. These criteria can be executed with BASIC or Machine Language porgrams. Since we are primarily programming in BASIC, we will develop some BASIC routines to show how the character can be produced and moved on the C1P's monitor screen.

The BASIC routine in section one of Listing 1 will be used to generate an animated character that will primarily move from near center screen downward to near the bottom of the screen. This subroutine begins at BASIC progran line 5. Here the REM statement tells the user that this is a routine to generate the movement of a character downward. Line 10 is the real beginning of the subroutine. At line 10 the A variable is loaded with the decimal beginning of the memory location where the character will first be displayed. Notice that this line forms part of a FOR/NEXT loop. Also notice that this loop will be incremented by a total of 32 counts for every pass through the program. This is done with the STEP function of BASIC. The FOR/NEXT loop at line 10 actually sets the limits of movement of the character. These limits are in the

For an angular movement in any direction,use the value in the chart to cause movement in that direction. It must be understood that the decimal beginning and ending must be caluclated because for each pass through the loop with a step function, the variable will be incremented by the amount in the step value.

Study the remainder of the modules in Listing 1, from our discussion you should be able to see just how these subroutines work. Load the programs into your C1P and watch the action on the screen. This will show you the results. The diagram in Figure 2 gives the complete memory map for a C1P. This is for a 25 by 25 character format. Use this diagram for all your plotting to find any location on the C1P monitor screen.

Now that we have seen some examples of how a moving character is made to move on the C1P's screen, let's use some of these techniques to develop a program that has some moving character elements that form a game. Listing 2 shows a game program that uses moving elements. These are: a starship and lasar cannon shots directed at the starship. All the techniques that we have discussed, that give the sensation of motion, are used in Listing 2. Please refer to this listing as we discuss the inner workings of the program's operation.

The program is presented, as I have said, as a game. The starship moves across mid-screen and the cannons are placed at each bottom corner, and at mid-bottom of the screen. The keyboard keys 5,6, and 7 are used to fire the cannons. A hit score total is printed out at the top of the monitor screen for the player.

This program is straight-forward and each module is identified by REM statements. This discussion will deal mainly with graphics and the keyboard routines, so please continue to refer to Listing 2. The remainder of the program should be self-explanatory.

The program from line 300 through 347 forms the main line BASIC module. It is used to draw and move the starship across the screen. The polling routine for the keyboard is located from line 335 to line 344. If a 5,6, or 7 key is pressed, a GOSUB to a cannon shot routine will result in a shot at the starship. Key 5 causes a shot from bottom right upward diagonally to top left of the screen. A 7 key results in a shot from bottom left to top right. A 6 key results in a true vertical shot.

The position of the starship is always contained in the K variable. This location is always checked in each shot routine at lines 415, 462, and 525. If a hit occurs, the program jumps to line 600 where an explosion of the starship will be displayed at the screen location contained in variable K. Next a hit score will be placed on the screen. The hit count will be checked for 10 hits. If so the player will be informed that he has completed the exact number of hits and has won the game. If the player has less than 10 hits, the program returns through RETURNs to the exact main-line program at line 300.

This program uses more of the elements contained in the Character Generator ROM. These elements are the elements that are used to draw the starship. Their decimal equivalents are 9 and 12, and are written into video memory with the POKE statement at line 310. After a delay at line 320, the starship is erased and placed at the next location in the FRO/NEXT loop from line 300. The cannon shots are primarily POKEd to screen memory, displayed for some duration of time and then erased. This process continues until the FOR/NEXT loop has been incremented to its maximum value.

### Conclusion

If you have followed all five parts of this series, I believe that you should now have sufficient knowledge of your C1P's graphics capabilities. I hope that you now also have a better understanding of the polled keyboard, and how to use these capabilities with BASIC programming to produce real working programs that will be enjoyable to use. Hopefully you have learned with me through these efforts and I will see some of your programs published in the pages of MICRO in the near future. With that, I will conclude this series of articles and I hope that these programs and ideas will be as much fun for you as you read and experiment, as they have been for me in the writing. Good luck with your programming and with your writing.

```
SECTION 1)
5 REM MOVE CHARACTER DOWN
10 FOR A = 53776 TO 54160 STEP 32
20 POKE A, 161
30 FOR B = 1 TO 50 : NEXT B
40 POKE A , 32
50 NEXT A

SECTION 2)
60 REM MOVE CHARACTER UP
70 FOR A = 54160 TO 53763 STEP- 32
80 POKE A, 161
90 FOR B = 1 TO 50 : NEXT B
100 POKE A , 32
110 NEXT A

SECTION 3 )
120 REM MOVE CHARACTER RIGHT
130 FOR A = 53776 TO 53787
140 POKE A , 161
150 FOR B = 1 TO 50 : NEXT B
160 POKE B , 32
170 NEXT A

SECTION 4 )
180 REM MOVE CHARACTER LEFT
190 FOR A = 53776 TO 53763 STEP -1
200 POKE A , 161
210 FOR B = 1 TO 50 : NEXT B
220 POKE A , 32
230 NEXT A
```

**List 1**

*Photographs for this series were provided by William L. Taylor, Jr.*

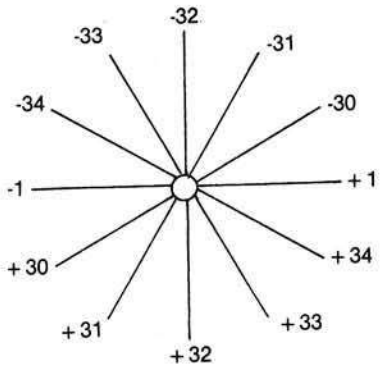



***Figure 1:* Plotting Directions for the C1P**

```
LIST1-500

 1 REM DEMONSTRATION PROGRAM FOR ANIMATED ELEMENTS ON
 2 REM BY WILLIAM L. TAYLOR 12/3/1979        OSI C1P
 3 PRINT"STAR SHIP ATTACK"
 4 PRINT:PRINT" DESTROY THE STARSHIPS WITH KEYS 5,6,7"
 5 PRINT" YOU GET 10 SHOTS":PRINT
 8 FOR S=1 TO 10000:NEXT S
 10 L=2:GOTO350
 299 REM DRAW STARSHIP AND KEYBOARD POLLING ROUTINE
 300 FOR K=53763 TO 53787
 310 POKE K,9:POKE K+1,12
 320 FOR J=1 TO 50:NEXT J
 330 POKE K,32:POKE K+1,32
 335 POKE 530,1:POKE 57088,127
 337 C=C+1
 342 IF PEEK(57088)=253 THEN GOSUB 400
 343 IF PEEK(57088)=251 THEN GOSUB 500
 344 IF PEEK(57088)=247 THEN GOSUB450
 345 NEXT K
 347 GOTO 300
 349 REM CLEAR SCREEN
 350 FOR T=-3 TO 32:PRINT:NEXT T
 360 GOTO 300
 399 REM DRAW RIGHT VERTICAL SHOT
 400 FOR T1=54147 TO 53403 STEP -31
 410 POKE T1,249
 415 IF T1=K THEN GOSUB 600
 420 FOR T2=1 TO 5:NEXT T2
 425 POKE T1,32
 430 NEXT T1
 438 RETURN
 449 REM DRAW LEFT VERTICAL SHOT
 450 FOR T3=54171 TO 53379 STEP -33
 460 POKE T3,255
 462 IF T3=K THEN GOSUB 600
 463 FOR T4=1 TO 5:NEXT T4
 464 POKE T3,32
 465 NEXT T3
 470 RETURN
 475 POKE T4,32
 499 REM DRAW VERTICAL SHOT
 500 FOR T5=54158 TO 53390 STEP-32

OK
LIST4_500-800

 500 FOR T5=54158 TO 53390 STEP-32
 520 POKE T5,248
 525 IF T5=K THEN GOSUB 600
 530 FOR T7=1 TO 5:NEXT T7
 540 POKE T5,32
 550 NEXT T5
 580 C=0:RETURN
 599 REM CHECK SHOT HIT DRAW EXPLOSION AND DISPLAY HITS
 600 E=0:U=53731
 610 POKE U+E,32
 620 FOR A=1 TO 10:POKE K,A:NEXT A
 630 E=E+1:IF E<20 THEN 610
 640 IF E=20 THEN POKE 53455,72:POKE 53456,73:POKE 53457,84
 650 POKE 53458,32:POKE 53459,L+47
 660 L=L+1
 662 IF L<10 THEN RETURN
 665 IF L=10 THEN PRINT" ALL TARGETS DESTROYED"
 670 PRINT" YOU HAVE SAVED THE UNIVERSE"
 680 PRINT" WANT TO PLAY AGAIN  YES OR NO"
 690 INPUT A1$
 700 IF A1$="YES" THEN 1
 710 IF A1$="NO"THEN END
OK
```


## OHIO SCIENTIFIC
## C1-P MINI-FLOPPY EXPANSION ACCESSORIES

| No. | Item | Quantity | Price Each |
|---|---|---|---|
| 0 | Challenger 1P **(Retail $349)** with manuals, demo program tape. Available with 4K or no memory (OK)L | | (OK) **$319.** (4K) $349 |
| 1 | Model 600 Superboard **(Retail $279)** with manuals, demo tape. Available with 4K or no memory | | (OK) **$249.** (4K) $279 |
| 2 | Model 610 Mini Floppy Expansion Board (expandable to 24K). Available with 8K or no memory. | | (OK) **$210.** (8K) $295 |
| 3 | Model 620 C1-P to OSI 48-pin C3 Bus Converter (optional) | | $ 25 (estimate) |
| 4 | 5V 3A Power Supply (for either 600 or 610 Board) Fits cabinet, item 10 directly | | $ 38.95 |
| 5 | MPI 40-track Disk Drive **(Retail $450)** in Metal Cabinet, with or without power supply (12V). | | no P/S $410. with P/S $450 |
| 5a | MPI 40 - track Disk Drive (no cabinet). Includes data separator | | $320 |
| 6 | Cable Adapter, 610 board to disk drive. Includes interface schematic | | $ 34.50 |
| 7 | Cable Adapter, 600 board to 610 board | | $ 9.95 |
| 8 | 4K Memory (8-2114's) | | $ 64. |
| 8a | 8K Memory (16-2114's) | | $110. |
| 9 | Sprite Fan (Needed for 12K and up) | | $ 20. |
| 10 | Challenger 1P Cabinet w. power cord and fuse | | $ 49.95 |
| 11 | Manuals Set, Challenger 1. Includes schematics | | $ 5.00 |
| 12 | Manual, MPI Disk Drive | | $10.00 |
| 13 | Manual, Shugart SA-400 Disk Drive | | $10.00 |
| — | Program Cassettes (Blank) 10 min. Best Quality. | | $ 1.00 |
| — | Programs (Graphics, Business, Home, etc.) | | $ 6.00 |
| Memory Capacity: 600 Superboard: 8K Total; 610 Expansion: 24K Total; Grand Total: Max 32K | | | Sub Total; 6% Tax; Total |

Please copy this list and use as order form. WE PAY SHIPPING

Check, Visa, Mastercharge
POSTPAID in USA.
California add 6% tax

## OHIO SCIENTIFIC EXPANSION INFORMATION

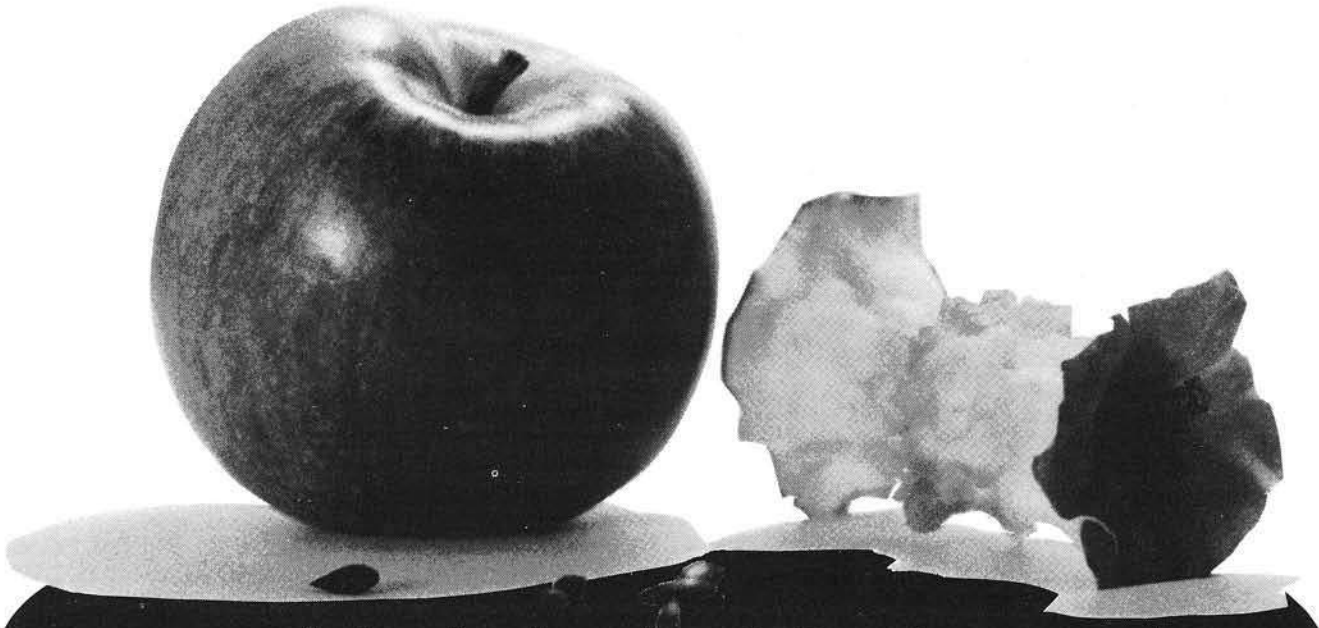
Conversion of C1P (Cassette) to 52x26 display. Detailed step-by-step instructions for doubling the C1P speed and display size!
Order Bulletin 1105 **$12.00**

Conversion of C1P (Minifloppy) to 52x26 display. Same as above but includes display driver for disk operating system.
Order Bulletin 1105MF **$20.00**

Conversion of C2-4P or C2-8P to 8 in. or 5¼ in. floppy. Instructions and schematics for converting cassette-based units into floppy systems.
Order Bulletin 1106F **$20.00**

- Ohio Scientific Challenger
- Commodore KIM
- S-44 Microprocessor Boards
- Expansion, Repair, and Software

**SILVER SPUR**
**Electronic Communications Co.**
3873F Schaefer Avenue
Chino, Calif. 91710
(714) 627-9366



# Introducing AppleSeed, our newest publication to whet your Apple* appetite!

We invite you to subscribe to AppleSeed - the magazine that is to the Apple II* what SoftSide is to the TRS-80**. It offers the newest in software programming hints and ideas tailored especially for your computer. AppleSeed features challenging programs for both the do-it-yourselfer and the individual interested in pre-packaged programs and games . . . your own preview of the best available on the market today. A typical slice of AppleSeed consists of one major (new 16K) commercial level program (completely listed for your keying pleasure), accompanied by two or three applications for practical use or fun, supplemented by informative articles to polish your Apple*. Get right to the core of your Apple* needs and order AppleSeed today! 12 issues, 1 year, $15.00. AppleSeed is the newest member of . . .

SoftSide™
PUBLICATIONS
6 South Street, Milford, NH 03055
(603) 673-5144

*A registered trademark of Apple Computers. **A registered trademark of Radio Shack and Tandy Corp.

# Lower Case and Punctuation in APPLESOFT

**Do you need to get lower case and punctuation into your BASIC strings? Then, try these programs.**

**James D. Childress**
**5108 Springlake Way**
**Baltimore, MD 21212**

### Introduction

While computer people may adapt to all caps, the general public still uses, and apparently likes, lower case. Printing with lower case is more familiar, more readable and more acceptable. Thus, we who work with computers should provide lower case in any printout that we expect or hope laymen to read. After all, computers should adapt to people; people should not have to adapt to computers.

Also, who is there among us who hasn't wondered at how the APPLE handles punctuations in strings? In INPUT's, we have found to our dismay that a "JONES, JOHN" results in an error message saying "?EXTRA IGNORED" and later finding the string variable as only "JONES" with nothing to tell us which Jones that may be. What wouldn't we give to get quotation marks and commas in the places we want?

So much for what should be or what we want. The APPLE doesn't have lower case and seems rather whimsical about punctuation. Well, face it; there were a number of compromises made in the design of the APPLE and Applesoft. Of course, some of these deficiencies can be conquered by money. We can buy one of the lower-case boards and live more or less happily ever after. Unfortunately, we do not all or always have the option of buying a solution to a problem; most of us have more problems than money. And there are not always solutions for sale.

An alternative approach is an Applesoft program to produce the desired lower case and punctuation. I have looked for such a program and I found two possibilities (there likely are others but I am not acquainted with them):

1. Val J. Golding in "Lower Case Routine for Integral Data Printer," *Call-Apple*, v.2, p. 11 (April/May 1979) gave a program to poke lower case characters into strings in the string array memory space.

2. Another program was published in *Contact*, v.1, p.5 (May 1978); this program pokes lower case into the beginning of program memory space.

Both of these are quite limited. Note: both should work for punctuation problems within the same limitations.

Neither of these enables one to enter lower case or problem punctuations conveniently into string variables, nor to print statement strings in an Applesoft program as desired. The program given in the listing in Figure 1 does the job for string variables and the one given in Figure 2, for strings in print statements.

### Use and Operation

The heart of these programs is the same as in the cited programs: use of the GET command to sneak things around the interpreter. The GET command handles input character by character so that each can be manipulated.

(The identical GET routine is used for both programs—lines 63010 to 63120 in the first and lines 63140 to 63150 in the second. Only one typing needs be done, a hint not to be ignored.)

The first program is intended for use as a subroutine. For example, a statement such as

```
30 INPUT "ACCOUNT NAME";NAME$(1)
```

can be replaced directly by

```
30 PRINT "ACCOUNT NAME";: GOSUB
    63000:NAME$(1) = BB$
```

In a run, the program would appear to behave normally except that there would be no ?EXTRA IGNORED's and NAME(1) would look quite strange on the CRT monitor (",/7%2 #!3%" for "lower case") and as lower case only on the printer.

In both programs, capitals are entered in a manner similar to the operation of MUSE's word processor program Dr. Memory. A ctrl-A makes the next letter only captial; an ctrl-C makes all the following letters capital until either a ctrl-S or the end of the string. Unlike Dr. Memory, the control characters are not displayed. Instead, the capitalized letters are shown in inverse video. I like this way of doing things. If you would prefer the opposite video, just interchange the words NORMAL and INVERSE in lines 63020-63040 and 63080 and add an INVERSE to line 63000 in Figure 1. You could do even more to tailor to your personal tastes; change the control characters, change the default operation from lower case to capitals, etc. These custom fittings are left as an exercise.

Another feature common to both programs is the motion of the cursor. The backspace works but that is all. And it will move the cursor back no further than the initail position. However, therein lurks a minor nuisance; if you try to backspace beyond that limit, the immediately preceeding character will be wiped out or replaced by a white block. This is of no consequence; ignore it.

Since the string variables subroutine runs as a part of your program, you have to keep labels straight. This subroutine uses only AA$, AZ$, BB$, BB, BZ$, and ZZ and has no FOR loops. Also note that only the usual limitation applies for the length of strings.

In the use of the second program, you append it to the program in which you

want to put lower case. A RUN 63000 initiates things; you simply give the line number in which lower case is wanted. The first string in that line is printed, terminated by ## to indicate the length limit. The cursor below this line indicates the place for the change. You can insert anything but we assume that a mixed capital and lower case rendition of the line above is what you will want. In any case, the length cannot be exceeded. If you go over the limit, the excess will be ignored. If you put in less, the remainder will be filled with spaces. If you don't want to change that particular string, simply hit RETURN.

### *Figure 1*

```
63000 BB$ = "":BZ$ = "":BB = 0:ZZ
      = 0
63010  GET AA$:AZ$ = AA$: IF  ASC
     (AA$) = 13 THEN  NORMAL : GOTO
     63130
63020  IF  ASC (AA$) = 1 THEN ZZ =
     1: INVERSE :BB = 0: GOTO 630
     10
63030  IF  ASC (AA$) = 3 THEN BB =
     1: INVERSE : GOTO 63010
63040  IF  ASC (AA$) = 19 THEN BB
      = 0: NORMAL : GOTO 63010
63050  IF ZZ = 1 OR BB = 1 THEN Z
     Z = 0: GOTO 63080
63060  IF  ASC (AA$) < 65 OR  ASC
     (AA$) > 90 THEN 63080
63070 AA$ =  CHR$ ( ASC (AA$) + 3
     2)
63080 BZ$ = BZ$ + AZ$: PRINT AZ$;
     : IF BB = 0 THEN  NORMAL
63090 BB$ = BB$ + AA$: IF  ASC (B
     B$) = 8 AND  ASC (AA$) = 8 THEN
      PRINT " ";
63100  IF  LEN (BB$) <  = 2 AND  ASC
     (AA$) = 8 THEN BB$ = "":BZ$ =
     "": GOTO 63010
63110  IF  ASC (AA$) = 8 THEN BB$
      =  LEFT$ (BB$, LEN (BB$) -
     2)
63120  GOTO 63010
63130  PRINT : RETURN
63140  END
```

### *Figure 2*

```
62999  END
63000  HOME : VTAB (3): PRINT "LO
     WER CASE INSERTION PROGRAM":
      PRINT : PRINT
63010 LMAX = 62999: PRINT "NUMBER
      OF FIRST LINE TO BE RE-": INPUT
     "WRITTEN  ";LT: PRINT
63020  PRINT :M = 256 *  PEEK (10
     4) +  PEEK (103) + 2
63030 LN = 256 *  PEEK (M + 1) +
      PEEK (M): IF LN >  = LMAX OR
     LN > LT THEN 63320
63040  IF LN <  > LT THEN M = 256
      *  PEEK (M - 1) +  PEEK (M -
     2) + 2: GOTO 63030
63050 K = 0:LL = 0:UL = 0
63060  FOR J = M + 2 TO M + 255:T
     ST =  PEEK (J): IF TST = 0 THEN
     M = J + 3: GOTO 63030
63070  IF TST = 58 THEN K = 0
63080  IF TST = 186 OR TST = 132 THEN
     K = 1
63090  IF K = 1 AND LL > 0 AND TS
     T = 34 THEN UL = J - 1: GOTO
     63120
63100  IF K = 1 AND LL = 0 AND TS
     T = 34 THEN LL = J + 1
63110  NEXT
63120 BB$ = "":BZ$ = "":BB = 0:ZZ
      = 0
63130  FOR I = LL TO UL: PRINT  CHR$
     ( PEEK (I));: NEXT : PRINT "
     ##"
63140  GET AA$:AZ$ = AA$: IF  ASC
     (AA$) = 13 THEN  NORMAL : GOTO
     63260
63150  IF  ASC (AA$) = 1 THEN ZZ =
     1: INVERSE :BB = 0: GOTO 631
     40
63160  IF  ASC (AA$) = 3 THEN#BB =
     1: INVERSE : GOTO 63140
63170  IF  ASC (AA$) = 19 THEN BB
      = 0: NORMAL : GOTO 63140
63180  IF ZZ = 1 OR BB = 1 THEN Z
     Z = 0: GOTO 63210
63190  IF  ASC (AA$) < 65 OR  ASC
     (AA$) > 90 THEN 63210
63200 AA$ =  CHR$ ( ASC (AA$) + 3
     2)
63210 BZ$ = BZ$ + AZ$: PRINT AZ$;
     : IF BB = 0 THEN  NORMAL
63220 BB$ = BB$ + AA$: IF  ASC (B
     B$) = 8 AND  ASC (AA$) = 8 THEN
      PRINT " ";
63230  IF  LEN (BB$) <  = 2 AND  ASC
     (AA$) = 8 THEN BB$ = "":BZ$ =
     "": GOTO 63140
63240  IF  ASC (AA$) = 8 THEN BB$
      =  LEFT$ (BB$, LEN (BB$) -
     2)
63250  GOTO 63140
63260  IF BB$ = "" THEN 63310
63270  PRINT : FOR I = LL TO UL
63280 DD$ =  MID$ (BB$,I - LL + 1
     ,1):MM =  ASC (DD$)
63290  POKE I,MM
63300  NEXT
63310 UL = 0:LL = 0: PRINT : GOTO
     63110
63320  PRINT : PRINT "NUMBER OF N
     EXT LINE TO BE REWRITTEN": INPUT
     "(ENTER 0 TO END PROGRAM)  "
     ;LT
63330  IF LT = 0 THEN  END
63340  GOTO 63020
```

After a RETURN, the next string in the same line will appear, ready to be changed. When all the strings of that one line have been dealt with, you are asked for the number of the next line.

As mentioned above, lower case if displayed by the APPLE ad keyboard symbols other than letters. These print properly as lower case on a printer that prints lowercase. If you want to display, say, a table so that you can check data prior to printing, you need to program the display table and the printout table seperately. For convenience in doing this, both programs provide an all-caps string BZ$ ad well as the corresponding string BB$ with lowercase.

### Program Design

The GET routine, essentially the whole of Figure 1, has already been mentioned. The GET command is follwed by a series of IF's to implement the control character,backspace and RETURN functions. These are straight-forward and self-explanatory.

The second program, Figure 2, consists of three parts. The first, lines 63020-63300, pokes the new string into the program into the program memory space.

### Concluding Remarks

Although written for Applesoft, these programs can be adapted to other BASIC's.The first presents no problems. However, the program memory space search routine in the second will require modification for other computers. This modification should not be too difficult to implement for other Microsoft BASIC' s.


# COMPUTER EQUIPMENT & SOFTWARE BARGAINS

COMPUTER SHOPPER
The Nationwide Marketplace for Computer Equipment
Get Your First Issue FREE!
TRS-80's at Super Discount Prices

**EVERY MONTH**

BUY, SELL OR TRADE ALL TYPES OF COMPUTER EQUIPMENT AND SOFTWARE (pre-owned and new) among 20,000 readers nationwide.

FEATURES:

- Low classified ad rates - 10¢ a word
- Hundreds of ads from individuals
- Categorized ads so you can find them instantly
- Large (11 by 14") easy to read pages

Subscribe now for $10 and receive 13 issues/year (one FREE plus 12 regular issues). After receiving your first issue if you're not completely satisfied you may have a 100% refund and you still keep the first issue free. Bank cards accepted.

BONUS: *If you have something to advertise (pre-owned or software) send in a classified ad with your subscription and we'll run it FREE.*

*The Nationwide Marketplace for Computer Equipment*

COMPUTER SHOPPER

*P.O. BOX F16 • TITUSVILLE, FL 32780 • 305-269-3211*



# Educational Software Professionals

**Apple-Grammer** Written by Tom Ankofski. This is a program which utilizes the basic skills of grammar. Nouns, verbs, pronouns, adjectives, adverbs, prepositions and conjuctions are presented in sentence—quiz form to the student. The program allows the teacher to choose specific areas of grammar that need attention and to alter the quiz easily. It provides positive reinforcement to the child by correcting his work in color graphics and scoring at the end. This program was a winner of ADS and NCC (National Computer Convention).Requires minimum of 16K Applesoft.

Tape $14.95
Disk 19.95

**Study Quiz Files** This self-explanatory program allows you to create and run study quizzes, save them on disk and retrieve tham at will. Many quizzes can be stored on this one disk. Review your items, revise them, add items, delete items without any computer programming knowledge. Written by a liscenced psychologist, the program includes ramdom question presentation, reinforcement of correct answer, with a display of the student's name, immediate right answer feedback after in correct responses, color congratulation display at the end and a final score summary. Guided instructions are easy to understand and follow even the first time around. Pupils can work independently selecting the appropriate quiz and running it with little or no supervision. Requires 32K Applesoft RAM or ROM.

Disk $19.95

**MULTIPLE CHOICE FILES** Same program as Study Quiz Files but allows different quiz presentation.

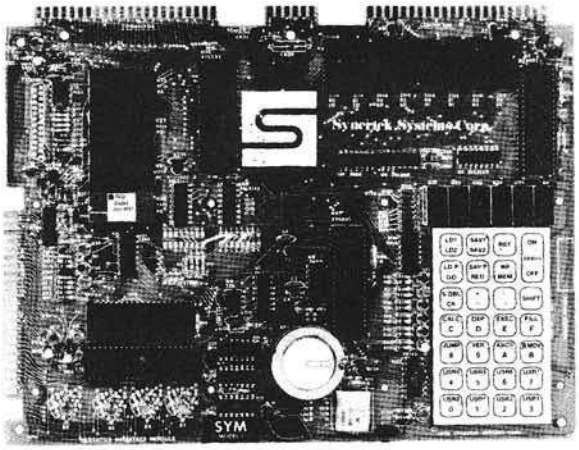
Disk $19.95

Postage and handling per order: $1.00

**ESP**
**38437 Grand River**
**Farmington Hill, MI 48108**

## SYM-1, 6502-BASED MICROCOMPUTER

- FULLY-ASSEMBLED AND COMPLETELY INTEGRATED SYSTEM that's ready-to-use
- ALL LSI IC'S ARE IN SOCKETS
- 28 DOUBLE-FUNCTION KEYPAD INCLUDING UP TO 24 "SPECIAL" FUNCTIONS
- EASY-TO-VIEW 6-DIGIT HEX LED DISPLAY
- KIM-1* HARDWARE COMPATIBILITY
  The powerful 6502 8-Bit MICROPROCESSOR whose advanced architectural features have made it one of the largest selling "micros" on the market today.
- THREE ON-BOARD PROGRAMMABLE INTERVAL TIMERS available to the user, expandable to five on-board.
- 4K BYTE ROM RESIDENT MONITOR and Operating Programs.
- Single 5 Volt power supply is all that is required.
- 1K BYTES OF 2114 STATIC RAM onboard with sockets provided for immediate expansion to 4K bytes onboard, with total memory expansion to 65, 536 bytes.
- USER PROM/ROM: The system is equipped with 3 PROM/ROM expansion sockets for 2316/2332 ROMs or 2716 EPROMs
- ENHANCED SOFTWARE with simplified user interface
- STANDARD INTERFACES INCLUDE:
  —Audio Cassette Recorder Interface with Remote Control (Two modes: 135 Baud KIM-1* compatible, Hi-Speed 1500 Baud)
  —Full duplex 20mA Teletype Interface
  —System Expansion Bus Interface
  —TV Controller Board Interface
  —CRT Compatible Interface (RS-232)
- APPLICATION PORT: 15 Bi-directional TTL Lines for user applications with expansion capability for added lines
- EXPANSION PORT FOR ADD-ON MODULES (51 I/O Lines included in the basic system)
- SEPARATE POWER SUPPLY connector for easy disconnect of the d-c power
- AUDIBLE RESPONSE KEYPAD




Synertek has enhanced KIM-1* software as well as the hardware. The software has simplified the user interface. The basic SYM-1 system is programmed in machine language. Monitor status is easily accessible, and the monitor gives the keypad user the same full functional capability of the TTY user. The SYM-1 has everything the KIM-1* has to offer, plus so much more that we cannot begin to tell you here. So, if you want to know more, the SYM-1 User Manual is available, separately.

| | |
|---|---|
| **SYM-1 Complete w/manuals** | **$229.00** |
| **SYM-1 User Manual Only** | **7.00** |
| **SYM-1 Expansion Kit** | **60.00** |

Expansion includes 3K of 2114 RAM chips and 1-6522 I/O chip.

SYM-1 Manuals: The well organized documentation package is complete and easy-to-understand.

SYM-1 CAN GROW AS YOU GROW. Its the system to BUILD-ON. Expansion features that are soon to be offered:

| | |
|---|---|
| ***BAS-1 8K Basic ROM (Microsoft)** | **$89.00** |
| ***KTM-2 TV Interface Board** | **319.00** |

**We do honor Synertek discount coupons*

## QUALITY EXPANSION BOARDS DESIGNED SPECIFICALLY FOR KIM-1, SYM-1 & AIM 65

These boards are set up for use with a regulated power supply such as the one below, but, provisions have been made so that you can add onboard regulators for use with an unregulated power supply. But, because of unreliability, we do not recommend the use of onboard regulators. All I.C.'s are socketed for ease of maintenance. *All boards carry full 90-day warranty.*

All products that we manufacture are designed to meet or exceed industrial standards. All components are first qualtiy and meet full manufacturer's specifications. All this and an extended burn-in is done to reduce the normal percentage of field failures by up to 75%. To you, this means the chance of inconvenience and lost time due to a failure is very rare; but, if it should happen, we guarantee a turn-around time of less than forty-eight hours for repair.

*Our money back guarantee:* If, for any reason you wish to return any board that you have purchased directly from us within ten (10) days after receipt, complete, in original condition, and in original shipping carton; we will give you a complete credit or refund less a $10.00 restocking charge per board.

### VAK-1 8-SLOT MOTHERBOARD

This motherboard uses the KIM-4* bus structure. It provides eight (8) expansion board sockets with rigid card cage. Separate jacks for audio cassette, TTY and power supply are provided. Fully buffered bus.

**VAK-1 Motherboard** **$139.00**

### VAK-2/4 16K STATIC RAM BOARD

This board using 2114 RAMs is configured in two (2) separately addressable 8K blocks with individual write-protect switches.

| | |
|---|---|
| **VAK-2 16K RAM Board with only 8K of RAM ( ½ populated)** | **$239.00** |
| **VAK-3 Complete set of chips to expand above board to 16K** | **$125.00** |
| **VAK-4 Fully populated 16K RAM** | **$325.00** |

### VAK-5 2708 EPROM PROGRAMMER

This board requires a +5 VDC and ±12 VDC, but has a DC to DC multiplyer so there is no need for an additional power supply. All software is resident in on-board ROM, and has a zero-insertion socket.

**VAK-5 2708 EPROM Programmer** **$249.00**

### VAK-6 EPROM BOARD

This board will hold 8K of 2708 or 2758, or 16K of 2716 or 2516 EPROMs. *EPROMs not included.*

**VAK-6 EPROM Board** **$119.00**

### VAK-7 COMPLETE FLOPPY-DISK SYSTEM (See March Issue of Micro)

### VAK-8 PROTYPING BOARD

This board allows you to create your own interfaces to plug into the motherboard. Etched circuitry is provided for regulators, address and data bus drivers; with a large area for either wire-wrapped or soldered IC circuitry.

**VAK-8 Protyping Board** **$39.00**

### POWER SUPPLIES

ALL POWER SUPPLIES are totally enclosed with grounded enclosures for safety, AC power cord, and carry a full 2-year warranty.

### FULL SYSTEM POWER SUPPLY

This power supply will handle a microcomputer and up to 65K of our VAK-4 RAM. ADDITIONAL FEATURES ARE: Over voltage Protection on 5 volts, fused, AC on/off switch. Equivalent to units selling for $225.00 or more.

**Provides +5 VDC @ 10 Amps & ±12 VDC @ 1 Amp**
**VAK-EPS Power Supply** **$119.00**

**KIM-1* Custom P.S. provides 5 VDC @ 1.2 Amps and +12 VDC @ .1 Amps**
**KCP-1 Power Supply** **$39.00**

**SYM-1 Custom P.S. provides 5 VDC @ 1.4 Amps**
**VCP-1 Power Supply** **$39.00**

*KIM is a product of MOS Technology

RNB ENTERPRISES
INCORPORATED

2967 W. Fairmount Avenue
Phoenix AZ. 85017
(602)265-7564

master charge VISA

# MICRO™

**PO Box 6502**
**Chelmsford, Mass 01824**

617-256-5515

"The BEST of MICRO Volume 1" contains all of the important material from the first six issues of **MICRO** in book form.

"The BEST of MICRO Volume 2" contains all of the important material from the second six issues [#7 to 12] of **MICRO** in book form.

"ALL of MICRO Volume 2" is all six issues of Volume 2, issues 7 to 12, at a special reduced price for a limited time while supplies last.

Back Issues:

Issues 7 to 12: .....

Issues 13 on: .....

All payments must be in US dollars.
Make checks payable to: MICRO
Foreign payments in International Money Order or cash.

**Subscription:** One Year = 12 issues. Circle correct category and write amount in space provided.

| | | |
|---|---|---|
| **Surface:** | | |
| United States | $15.00 | |
| All Other Countries | $18.00 | |
| **Air Mail:** | | |
| Central America | $27.00 | |
| Europe/So. America | $33.00 | |
| All Other Countries | $39.00 | $ ..... |
| "BEST of MICRO Volume 1" | | |
| Surface | $7.00 | |
| Air Mail | $10.00 | $ ..... |
| "BEST of MICRO Volume 2" | | |
| Surface | $9.00 | |
| Air Mail | $13.00 | $ ..... |
| "ALL of MICRO Volume 2" | | |
| Surface | $9.00 | |
| Air Mail | $13.00 | $ ..... |

No. ..... Surface @ $1.75 each / Air Mail @$2.75 each = $ .....

No. ..... Surface @ $2.25 each / Air Mail @ $3.25 each = $ .....

**TOTAL** $ .....

If you are a subscriber, attach label or write subscription number here: .....

Name: .....

Address: .....

.....

City: ..... State: ..... Zip: .....

Country (if not U.S.): .....

Help **MICRO** bring you the info you want by completing this short questionnaire.

Microcomputers Owned/Planning to Buy: AIM SYM KIM PET APPLE OSI Other: .....

Peripherals Owned/Planning to Buy: Memory Disk Video Printer Terminal Other: .....

Microcomputer Usage: Educational Business Personal Control Games Other: .....

Languages Used: Assembler BASIC FORTH PASCAL Other: .....

Your comments and suggestions on **MICRO**: .....

.....

---

**Club/Group User Registration Form**

Name: ..... President: .....

Location: .....

No. of Members: .....

Meeting algorithm (date, time, place): .....

.....

.....

Publications: .....

.....

Aim/Purpose of the group: .....

.....

For Current Information, Contact: .....

.....

.....

### Software/Hardware Catalogue Entry

Do you have a software or hardware package you want publicized? Our Software and Hardware Catalogues offer a good opportunity to receive some free advertisement. These regular features of MICRO are provided both as a service to our readers and as a service to the 6502 industry which is working hard to develop new and better software and hardware products for the 6502 based system. There is no charge for listings in these catalogues. All that is required is that material for the listing be submitted in the listing format. All information should be included. We reserve the right to edit and/or reject any submission. We might not edit the description the same way you would, so please, be brief and specific.

Name:.....

System:.....

Memory:.....

Language:.....

Hardware:.....

Description:.....

.....

.....

.....

.....

.....

.....

Copies:.....

Price:.....

Author:.....

Available from:.....

.....

.....

### Classified Ad

Classified ads provide an economical way to announce new products or sales promotions, generate product interest, enhance visibility and promote good will. MICRO clusters large format classified ads at high impact locations throughout each issue. Because classifieds represent a service to readers, MICRO must restrict each advertiser to a single, six-line insertion per issue. The nominal $10 charge reflects our preparation costs and must be prepaid.

Description:.....

.....

.....

.....

.....

.....

.....

Name:.....

Address:.....

City,State:.....Zip: .....

### Other

If you are interested in Writing for MICRO, becoming a MICRO dealer or advertising in MICRO, please indicate below and the information will be mailed to you.

Dealer Information Package: ☐

Advertiser's Media Package: ☐

MICRO Writer's Guide: ☐

.....

First
Class
Stamp

MICRO™

**P.O. Box 6502**
**Chelmsford, MA 01824**

# SYM - 1 Sends Morse Code

**Now you can use your SYM as a Morse Code teaching tool, automatic I.D'er or 'canned' message sender.**

**Ralph R. Orton**
**16015 San Fernando Mission Blvd.**
**Granada Hills, CA 91344**

Although many Morse Code oriented programs have been written ranging from simple message loops to quite flexible code reading routines, I have not yet seen any written specifically for the SYM-1. The following will fill this gap with a sending program that could be used as a teaching tool, automatic I.D.'er or as a short cut for sending often sent messages. About 25 words can be stored with the 1K memory that comes with the SYM-1. An additional 50 words can be stored for each additional 1K memory added; thus, the 4K board R/W memory capability could store a total of about 175 words. This may not seem like a lot yet; teaching code at 5 wpm (words per minute), one would have over one half hour of steady material. Even at 13 wpm, you would have over 10 minutes of practice; no easy task for a learner! Figure 1 is a simple circuit for interfacing to the SYM-1 to provide an audio code indicator. Headphone jacks for several people could possibly be paralleled instead of the loud speaker. Other interfaces are left to the needs of the reader.

Pressing an 'O' on the keyboard enters a 'DIT' into memory. A '1' enters a 'dah' and a '2' enters a 'space' (enter 1 between letters and 3 between words). Spaces between parts of a letter need not be entered as they are provided by the program. Entering a '3' ends a message segment. This is only required if a series of messages are being entered. (See list of key memory locations.) As dits, dahs and spaces are entered from the keyboard on the SYM-1®'s, 1's and 2's appear on the display, indicating the data entered. Entry errors can be corrected by entering an 'E' for each entry to be erased. For example, if two erroneous entries had just been made, pressing 'E' twice would cause 'E' to be displayed twice. This indicates that the two prior entries had been erased (see figure 2). Upon completion of data entry, press the 'GO' key and your message will be sent.

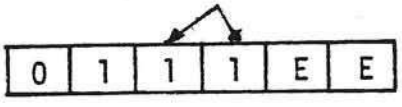
A popular method of teaching code is to send letters at a fast rate but leaving larger than normal spaces between letters until the learner has reached the desired plateau of proficiency.

The rate modification table can be used to determine data to be entered for desired combination of letter speed versus words per minute. Dit delay factor is entered at 0091 and the space factor is entered at 0076.

If continuous loop has been programmed at 004A through 004C, then code will be sent until such time reset is accomplished. If multiple message has been programmed, then a "GO" "CR" at the end of each segment will cause the nest segment to be sent.

It should be noted that a GO command at 002D will cause a new start to occur regardless of the mode selected. Thus, it is not necessary to reprogram old data unless it has been lost due to a newer entry.

**These Characters Have Been Erased**




04A thru 004C: These locations control the mode of operation.

4C 35 00 Gives continuous loop message. Be sure to put enough spaces at the beginning or end to identify the start of each loop through.

4C A2 00 Gives single or multiple message as desired. For multiple messages, key-in 'GO' 'CR' to start next message.

0053: Data at this location determines times Dit delay will be executed per 'Dit'.

0067: Data at this location determines times Dit delay will be executed per 'DAH'.

0076: Data at this location determines times Dit delay will be executed per 'Space'.

007D: Data at this location determines times Dit delay will be executed per silence between parts of a letter of spaces.

0091: Data at this location determines times delay programmed by "DIVFAC" (Division Factor) will be executed. (e.g. if 'Divfac' = 1024 then 1 loop = 1.024 ms disregarding instruction time error. Part of "Ditdly" routine.

0093: Data at this location determines divison factor to be used by internal timer. 1C = ÷ 1; 1D = ÷ 8; 1E = ÷ 64; 1F = ÷ 1024. Part of "Ditdly routine."

| Modified word rate | 13 | 17 | 21 | 25 | 29 |
|---|---|---|---|---|---|
| | Standard word rate | | | | |
| 5 | 0E | 14 | 1A | 20 | 27 |
| 8 | 06 | 0A | 0E | 11 | 15 |
| 11 | 02 | 05 | 08 | 0B | 0D |
| 14 | | 03 | 05 | 07 | 30 |
| 17 | | 01 | 03 | 05 | 06 |
| 20 | | | | 03 | 04 |
| 23 | | | | 02 | 03 |
| 26 | | | | | 02 |
| | Dit delay factor | | | | |
| | 5A | 45 | 38 | 2F | 28 |

RATE MODIFICATION TABLE

The timing in the table is based on the following relationships for standard code:

1. A dit is a reference unit of time.
2. A dah = 3 dits
3. Average letters = 6.2 dits
4. Spaces in a letter = 1 dit
5. Spaces between letters = 3 dits
6. Words = 5 letters & appropriate spaces

"Space" multiplication factor =

$$\frac{60 + d - d\,(43\,Wm - 3)}{d(7W - 3)}$$

Where d = dit time of standard words per minute rate
Wm = words per minute of the desired modified rate.
"dit time" =

$$\frac{60}{50Ws - 7}$$

Where Ws = words per minute of the desired standard rate.

The above formulas neglect the operation times of the SYM-1 but for practical purposes are quite accurate. The results must be converted to Hex for use in the program, introducing a rounding error which is also normally inconsequential. Greater accuracy is obtainable ofcourse, but the author leaves it to those with the desire to make the needed changes.

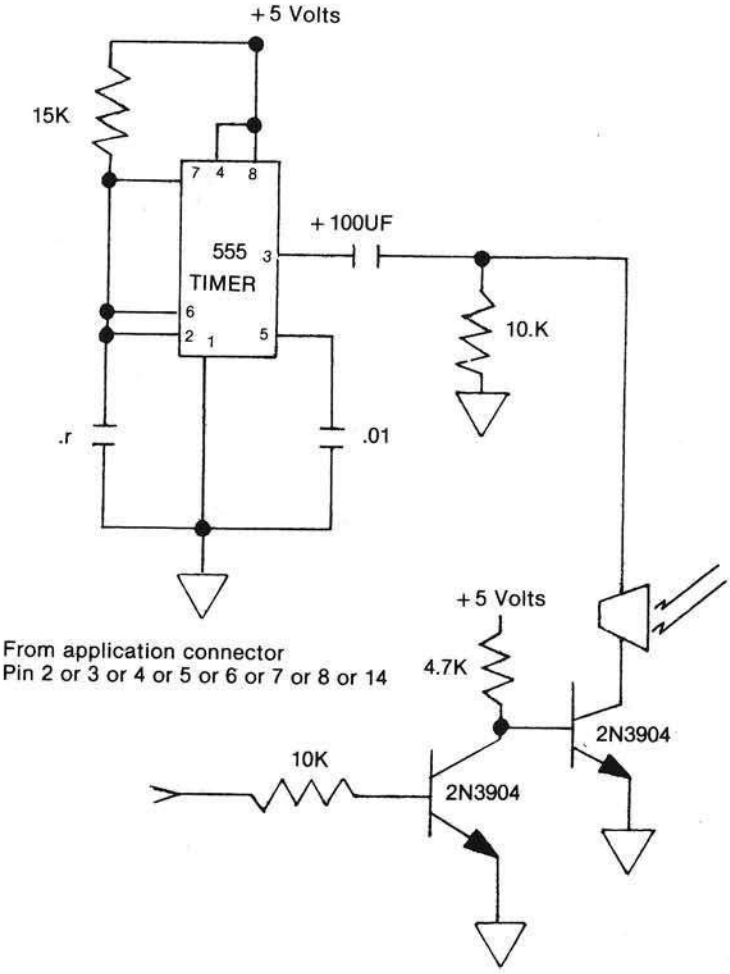



***Figure 1***

```
0090:                    EQUATE LIST
0100:
0110: 00A6               WORDS   *       $000A
0120: 00A6               INCHR   *       $8A1B
0130: 00A6               ACCESS  *       $8B86
0140: 0000                       ORG     $0000
0150:
0160: 0000 A0 00         LOAD    LDYIM   $00
0170: 0002 20 86 8B              JSR     ACCESS
0180: 0005 20 1B 8A      SHOKEY  JSR     INCHR
0190: 0008 99 00 02              STAY    $0200   WORDS ARE STORED STARTING AT $0200
0200: 000B C9 47                 CMPIM   $47     WAS 'GO' KEY PRESSED ?
0210: 000D F0 1E                 BEQ     START   IF YES - START SENDING CODE
0220: 000F C9 45                 CMPIM   $45     WAS 'E' PRESSED ?
0230: 0011 F0 0D                 BEQ     ERASE   IF YES - DO ERASE ROUTINE
0240: 0013 C8                    INY
0250: 0014 C0 00                 CPYIM   $00     256 WORDS COMPLETED ?
0260: 0016 F0 03                 BEQ     BASELD  IF SO, INCREMENT HI BYTE OF BASE
0270: 0018 4C 05 00              JMP     SHOKEY
0280:
0290: 001B E6 0A         BASELD  INCZ    WORDS
0300: 001D 4C 05 00              JMP     SHOKEY
0310:
0320: 0020 88            ERASE   DEY
0330: 0021 C0 FF                 CPYIM   $FF     PAGE CROSSING ?
0340: 0023 F0 03                 BEQ     SUBASE  IF YES - DECREMENT HIGH BYTE OF BASE
```

```
0350: 0025 4C 05 00          JMP   SHOKEY
0360:
0370: 0028 C6 0A      SUBASE DEC   WORDS
0380: 002A 4C 05 00          JMP   SHOKEY
0390:
0400: 002D A9 FF      START  LDAIM $FF     START OF SENDING CODE
0410: 002F 8D 01 A0          STA   $A001
0420: 0032 8D 03 A0          STA   $A003
0430: 0035 A9 02      KEEPON LDAIM $02     'MODE' JUMPS HERE FOR LOOP
0440: 0037 85 3D             STAZ  $3D
0450: 0039 A0 00             LDYIM $00
0460: 003B B9 00 02   CODE   LDAY  $0200   CODE WAS STORED STARTING AT $0200
0470: 003E C9 30             CMPIM $30     IS IT A DIT ?
0480: 0040 F0 0B             BEQ   DIT     IF SO - GO TO DIT ROUTINE
0490: 0042 C9 31             CMPIM $31     IS IT A DAH ?
0500: 0044 F0 1B             BEQ   DAH     IF SO - GO TO DAH ROUTINE
0510: 0046 C9 32             CMPIM $32     IS IT A SPACE CHARACTER ?
0520: 0048 F0 2B             BEQ   SPACE   IF SO - GO TO SPACE ROUTINE
0530: 004A 4C A2 00   MODE   JMP   SEGMNT  NONE OF ABOVE , DECIDE MODE
0540:
0550: 004D A9 00      DIT    LDAIM $00
0560: 004F 8D 01 A0          STA   $A001   SET OUTPUT LOW
0570: 0052 A9 01             LDAIM $01     LOAD 1 FOR DIT DELAY
0580: 0054 85 FF             STA   $00FF   STORE FOR USE BY 'DITDLY'
0590: 0056 20 90 00          JSR   DITDLY
0600: 0059 A9 FF             LDAIM $FF
0610: 005B 8D 01 A0          STA   $A001   SET OUTPUT HIGH AGAIN
0620: 005E 4C 7C 00          JMP   SILENT
0630:
0640: 0061 A9 00      DAH    LDAIM $00
0650: 0063 8D 01 A0          STA   $A001   SET OUTPUT LOW
0660: 0066 A9 03             LDAIM $03     LOAD FOR 3 DIT DELAYS
0670: 0068 85 FF             STA   $00FF   STORE FOR USE BY 'DITDLY'
0680: 006A 20 90 00          JSR   DITDLY
0690: 006D A9 FF             LDAIM $FF
0700: 006F 8D 01 A0          STA   $A001   SET OUTPUT HIGH AGAIN
0710: 0072 4C 7C 00          JMP   SILENT  QUIET BETWEEN CHARACTERS
0720:
0730: 0075 A9 01      SPACE  LDAIM $01     LOAD $0076 FOR DESIRED SPACE
0740: 0077 85 FF             STA   $00FF      LENGTH AND STORE FOR USE BY 'DITDLY'
0750: 0079 20 90 00          JSR   DITDLY
0760:
0770: 007C A9 01      SILENT LDAIM $01     LOAD FOR 1 DIT DELAY
0780: 007E 85 FF             STA   $00FF   STORE FOR USE BY 'DITDLY'
0790: 0080 20 90 00          JSR   DITDLY
0800: 0083 C8         INCMEM INY           MOVE POINTER TO NEXT CHARACTER
0810: 0084 C0 00             CPYIM $00     PAGE CROSSING ?
0820: 0086 F0 03             BEQ   BASEGO  IF YES - INCREMENT HIGH BYTE BASE
0830: 0088 4C 3B 00          JMP   CODE
0840:
0850: 008B E6 3D      BASEGO INC   CODE    +02
0860: 008D 4C 3B 00          JMP   CODE    GET NEXT CHARACTER
0870:
0880: 0090 A9 47      DITDLY LDAIM $47     LOAD $0091 WITH DESIRED DIT TIME
0890: 0092 8D 1F A4   DIVFAC STA   $A41F
0900: 0095 2C 05 A4   TIMER  BIT   $A405
0910: 0098 10 FB             BPL   TIMER   KEEP CHECKING FOR DELAY COMPLETED
0920: 009A E8                INX           DONE - INCREMENT DIT COUNTER
0930: 009B E4 FF             CPX   $00FF   SHOULD WE DITDLY AGAIN ?
0940: 009D D0 F1             BNE   DITDLY
0950: 009F A2 00             LDXIM $00     RESET 'X' REGISTER
0960: 00A1 60                RTS           BACK TO WHERE YOU CAME FROM
0970:
0980: 00A2 00         SEGMNT BRK           STOP UNTIL "GO" "CARRIAGE RETURN
0990: 00A3 4C 83 00          JMP   INCMEM  NOW SEND NEXT MESSAGE
1000:
ID=
```

# SUPER-TEXT™

## STANDARD FEATURES

- single key cursor control
- automatic word overflow
- character, word and line insertion
- forward and backward scrolling
- automatic on screen tabbing
- single key for entering "the"
- auto paragraph indentation
- character, word and line deletion
- ditto key
- multiple text windows
- block copy, save and delete
- advanced file handling
- global (multi-file) search and replace
- on screen math and column totals
- column decimal alignment
- chapter relative page numbering
- complete printer tab control
- line centering
- superscripting and subscripting
- displays UPPER and lower case on the screen with Dan Paymar's Lower Case Adapter

# The Professional Word Processor

for the Apple II
and the Apple II plus

**MUSE**™
THE LEADER IN QUALITY SOFTWARE

## FAST EDITING

Super-Text was designed by a professional writer for simple, efficient operation. A full floating cursor and multiple text screens facilitate editing one section of text while referencing another. Super-Text's advanced features actually make it easier to operate, allowing you to concentrate on writing rather than remembering complicated key sequences.

## FLOATING POINT CALCULATOR

A built in 15 digit calculator performs on-screen calculations, column totals and verifies numeric data in statistical documents.

## EXCLUSIVE AUTOLINK

Easily link an unlimited number of on-line files on one disk or from disk to disk. Autolink allows you to search or print all on-line files with a single command. Typical files of items that can be stored in this way include personnel files, prospect files, maintenance records, training records and medical histories.

## ADVANCED FILE HANDLING

Single key file manipulation and complete block operations allow the user to quickly piece together stored paragraphs and phrases. Text files are listed in a directory with a corresponding index for fast and accurate text retrieval.

## PRINTER CONTROLS

Super-Text is compatible with any printer that interfaces with an Apple. Print single or multiple copies of your text files or link files and they will be automatically printed in the specified order. User defined control characters can activate most special printer functions.

## MODULAR DESIGN

This is a modularly designed system with the flexibility for meeting your future word processing needs. The first add-on module will be a form letter generator for matching mailing lists with Super-Text form letters.

SUPER-TEXT, requires 48K ($99.95)
Available TODAY at Computer Stores nationwide. Dealer inquiries welcome. For more information write:

MUSE SOFTWARE 330 N. Charles Street Baltimore, MD 21201 (301) 659-7212

# An EDIT Mask Routine in Applesoft BASIC

**This article describes some techniques for producing formatted output using Edit Masks. The programs permit you to produce professional looking output.**

Lee Reynolds
801 NE 18th Ct. 109
Ft. Lauderdale, FL 33305

My work as a professional programmer in business applications has often called for the use of what are called "edit masks", in such languages as COBOL, DIBOL, and the Commercial Subroutine Package of Data General FORTRAN. I have found the edit mask capability in these languages quite useful, and so I decided to write a routine in Applesoft Basic that I could use at home on my Apple II.

I should begin by first giving a brief explanation of what an edit mask is, for those readers who have never encountered the term before. An edit mask might be defined as a string of characters which specify operations on a number so as to produce an output string that contains the number's digits re-formatted for printing in certain specific ways. Some of the most common operations that can be carried out on any given number by means of edit masks are the following: (1) "suppressing" of zeroes, by replacing them with blanks in the output string, (2) inserting of a decimal point in a fixed position of the output string, (3) inserting of comma in the string to express thousands, million, etc., (4) placing a dollar sign before the leftmost digit of the number string, and (5) appending a minus sign to the end of the string if the input number is negative.

The edit mask is used as a sort of "picture" of what the output string should be like after carrying out operations such as the above on the number to be edited. In order to achieve this, there are defintie rules for the edit routine's interpretation of the characters that make up the mask. Perhaps the best way of explaining this is to give some examples of my routine's use.

The routine itself, on the following listing, is contained between line numbers 100 to 580. The statements preceding 100 are a "driver" routine you can use to input your edit mask and number to be editied in order to experiment with various types of editing.

The editing routine is called by means of a GOSUB 100. There are two arguments that must be passed to it: NUM is the number to be edited, and MASK$ is the edit mask string. NUM can contain any number of digits up to 9. I have made no provision for editing numbers that must be expressed in "scientific notation" with an Exponent field.

The result of the masking will be passed back to the calling program in the string OUT$, whose length is the same as MASK$.

There are six special characters which can appear in MASK$ that are treated in a distinctive way: these are the digit 9, the digit 0, the period, the comma, the minus sign, and the dollar sign. The mask can contain other characters also, but more about this later.

The digit 9 is the "numeric replacement" character. This means, wherever a 9 is present in the mask, it will be replaced in the result field (OUT$) by the corresponding digit of NUM, if any, in that position.

Thus, suppose we define MASK$ = "99999", and assume the number to be edited is NUM = 352. Then the result, after calling the edit routine, will be OUT$ = "352". (Note the two blanks preceding the ASCII digit 3. This is because the length of the mask exceeds the length of the number to edit by two.)

Next, the digit 0 is the "zero-suppress" character. This means wherever a 0 appears in the mask, it will be replaced in the result field by the corresponding digit of NUM only if that digit is not a zero; if the digit is a zero, then the corresponding position in the result field will be a blank.

To give an example, suppose MASK$ = "990990" and the number to be edited is NUM = 120563. Then the result will be OUT$ = "12 563". The zero in NUM was suppressed.

The most common usage of the zero-suppress character in a mask is to suppress leading zeroes of a number. Thus a mask of "00099" would suppress the first three digits of any five-digit number if they were zeroes, but would print them if they were not. Due to the way my routine operates, it turns out that leading zeroes are always suppressed, anyway. If you would rather change this feature of the routine, I will describe later how you could go about doing so.

The period in a mask is usually used as the decimal point position. It is what is called an "insertion character" in the mask because it is always inserted in the result field exactly in its corresponding position in the mask.

Let's consider some examples of masks containing a period, and what the result will be. Suppose our mask is "999.99", and our number to be edited is 312.44; then, as you would expect, the result will be OUT$ = "312.44". Next suppose we use the same mask but NUM = 33.6. The result is OUT$ = " 33.60". There is a blank in position one and a zero in the last position. (If the last character of the mask had been a 0 instead of a 9, then the last character in the result would have been a blank.) Now, let's suppose that NUM = 124.556. In this case there is one more digit to the right of the decimal point in the number to edit that there is in the decimal part of the mask. When this, or something similar happens, my routine will truncate the extra digit(s), and replace it (them) by an asterisk to signal

field overflow. The result then is OUT$ = "124.5*".

My routine follows a similar rule in case the number of digits to the left of the decimal point in NUM exceeds the number allowed in MASK$. For example, if NUM = 1256.7, then the result will be OUT$ = "*56.70".

By the way, since it is conceivable that you might, either by mistake or be design, include two or more periods in your mask, the routine will treat only the rightmost period in the mask as the decimal point position. All other periods will be treated as insertion characters, and so will appear in the corresponding positions of the result field as they expected.

Next, let's consider the comma in an edit mask. An example of a mask containing two commas is the following: MASK$ = "99,999,999". If your number to edit contains either 7 or 8 digits, then the result field will contain both commas in the appropriate places, as you would expect. However, with 6 or fewer digits in NUM, either the first or both commas will be suppressed and replaced by blanks. Examples: if NUM = 1234567, the OUT$ = " 1,234,567"; and if NUM = 1234, then OUT$ = "  1,234" (note the five blank characters preceding the digit 1); and lastly, if NUM = 123, then there will appear seven blanks preceding the digit 1: OUT$ = "  123".

Thus we see that the comma is a special sort of insertion character which is suppressed if there are no preceding digits of the number to be edited.

Now consider the dollar sign used as an edit mask character. I have defined this character's usage in a special way. IF the dollar sign is the very first character in the mask, then it is treated as what is called a "floating dollar signt". That means that the dollar sign in the result field will "float" to the right, far enough so as to immediately precede the leftmost digit of NUM. Some exaples: if MASK$ = "$99,999.99" and NUM = 11.45, then the result of editing is OUT$ = "  $11.45" (note that there are four blanks preceding the dollar sign in the result field). And if NUM = 2321, then we have this result: OUT$ = " $2,321.00" (one blank preceding the dollar sign).

Please note that I have defined this usage of the dollar sign as a "floating" dollar sign only when it is the first character in the mask. If it occurs elsewhere in the mask, then it becomes an insertion character.

The last special usage character in a mask is the trailing minus sign. If the mask contains a minus sign as the very last character, then the rightmost position of the result field will be a minus sign when the number to edit is negative, or will be blank if the number is positive. Examples: if MASK$ = "99,999.99-" and NUM = -1453.62, then the resultant OUT$ = " 1,453,62-". While if NUM = 2246.7, then we have OUT$ = " 2,246.70".

If a minus sign appears in a mask in any other position, it is treated as an insertion character. Thus, for example, you could format a date, MMDDYY = month, day, and year with the following mask: MASK$ = "09-99-99". If NUM = 101479, then OUT$ = "10-14-79".

You might be wondering what will happen if you edit a negative number using a mask which does not contain a trailing minus sign. It depends upon whether you have allotted enough digit positions in the mask to accommodate a leading minus sign. If you have then the minus sign will take the place of the first position containing a nine, zero, or comma that immediately precedes the leftmost digit of NUM. If you have not allotted enough digit positions in the mask, then my routine will print the asterisk signaling field overflow.

Now, any character other than the six special cases discussed above may also appear in a mask. In that case the character becomes an insertion character. Suppose you define

**MASK$ = "$BAL. DUE AS OF SEP/'78:**
**'99,999.99"**

If NUM = 1324.57, then the result of masking will be:

OUT$ = "BAL. DUE AS OF SEP/'78:
$1,324.57"

From the above example, you can see that you are only restricted in using edit masks by your imagination, perhaps after making modifications to my routine. For example, you will note that the year in the above mask is '78 not '79. It could not be '79 because the 9 is a numeric replacement character and in this case would have been blanked out. However, if you change the numeric replacement character to some other more convenient character (perhaps an ampersand?) then this difficulty could be avoided.

As already mentioned, another modification you might wish to make is to allow outputting of leading zeroes in a numeric field if the corresponding edit characters are 9's. To do this, you need to make three changes to the routine.

```
455 IF I-1 > =I1 AND MID$
                (MASK$,I-1,1) =
                "9" then 480
500 IF N$ = " " THEN N$ ="0"
525 IF N$ = " " THEN 460
```

When you incorporate this routine into your own programs, you may wish to change the names of some of the local variables used by it in order not to conflict with your own use of the same names. So here is a list of all variables used by my routine.

**Variables**

| | |
|---|---|
| MASK$ | the string containing the edit mask. |
| NUM | the input number to edit |
| NUM$ | NUM converted to a string |
| LM | length of MASK$ |
| LN | length of NUM$ |
| PM | position of rightmost decimal point in MASK$ (or zero if none) |
| PN | position of decimal point in NUM$ (zero if none) |
| RM | number of digit positions right of decimal point in MASK$ |
| RN | number of digits right of decimal point in NUM$ |
| QM | number of digit positions left of decimal point in MASK$ |
| QN | number of digits left of decimal point in NUM$ |
| FD% | flag telling whether mask has floating dollar sign (1 if yes, 0 if no) |
| MF% | flag telling whether mask has trailing minus sign (1 if yes, 0 if no) |
| NF% | flag telling whether NUM is negative (1) or positive (0) |
| M$ | current character of MASK$ being processed |
| N$ | current character of NUM$ being processed |
| I | loop variable and temporary variable |
| J | pointer to current digit in NUM$ |
| I1 | first position in MASK$ to process |
| I2 | last position in MASK$ to process |

One final note: in using the driver routine to experiment with various edit masks, you should remember that if your mask will contain commas or colons, then you must enclose the entire mask by quotation marks, or else Applesoft will drop part of your mask when it executes the INPUT statement.

**Notes on Converting to other Basics**

I am not familiar with any other Basics for microcomputers. I do, however, have some acquaintance with the Basic languages for two mini-computers—the DEC PDP-II and the Data General Nova 3. With this as background, I have compiled the following list of possible modifications you might have to make to my routine to get it to work on other 6502 machines other than the Apple.

1.) Applesoft allows variables to have names with more than two characters,

although only the first two are used to distinguish between between different names. If your Basic does not allow this, you will have to change some of the names that my routine uses.

2.) Some Basics don't allow multiple statements per line, or if they do, the statement separator might not be the colon; two common alternatives are the back slash or the exclamation point.

3.) If your Basic does not have the "ON...GO TO" statement, then line number 85 will have to be replaced with something else, perhaps a couple of "IF...THEN GOTO..." statements.

4.) Not all Basics allow "NEXT" statements which do not specify the loop variable to end "FOR" loops. There are several lines in my program that may necessitate this type of change: 160, 190, 240, 280, 340, and 550. In all of these cases the implied FOR loop variable is "I".

5.) You may have to DIMension your strings in your Basic program, as is true in Apple's Integer Basic, but not Applesoft.

6.) String concatenation in Applesoft is accomplished with string expressions joined by means of the plus (+) sign; your Basic may use the ampersand (&).

7.) In comparing strings, Applesoft uses the combination of less than and greater than signs (<>); perhaps, as in Integer Basic on the Apple, you are only allowed to test inequality with the number sign (#).

8.) Please note that I have several statements in my program of the following general form: IF X THEN... This is "shorthand" for the equivalent IF X <> 0 THEN... I also have a number of statements like the following: IF...THEN 100 (where 100 can be any statement number). This is a "shorthand" for IF...THEN GOTO 100. I don't know whether all Basics allow the abbreviated forms that I use.

9.) I have made use of the following string functions: STR$, LEFT$, RIGHT$, MID$, and LEN. Your Basic might call these by different names, or have different syntax rules about their arguments. Here are the Applesoft syntactic definitions for these functions, which you should keep in mind if you have to convert to different usages on your computer:

STR$(X)
: converts the number X to a string

LEFT$(A$,N)
: returns the leftmost N characters of string A$

RIGHT$(A$,N)
: returns the rightmost N characters of string A$

MID$(A$,M;N)
: returns the N consecutive characters of string A$, starting at position M

LEN(A$)
: returns the number of characters in string A$

These are all the differences between Applesoft and other Basics that I am aware of, although there may be more. At any rate, it should not be difficult to convert my program to any other machine's Basic.

```
]LIST

 10  REM  ROUTINE TO EDIT A NUMBER
     , NUM, WITH AN EDIT MASK, MA
     SK$
 20  HOME : PRINT "EDIT MASK ROUTI
     NE": PRINT : PRINT "   THE E
     DIT MASK CAN CONTAIN ANY INS
     ER-": PRINT "TION CHARACTERS
     , PLUS FOLLOWING SPECIAL"
 30  PRINT "CHARACTERS:": PRINT "
      IF $ IS FIRST CHAR., IT IS
     TREATED AS": PRINT "A FLOATI
     NG DOLLAR SIGN"
 40  PRINT "  IF - IS LAST CHAR.,
     IT WILL BE OUTPUT": PRINT "I
     F NUMBER TO EDIT IS NEGATIVE
     , OR RE-": PRINT "PLACED BY
     BLANK IF POSITIVE"
 50  PRINT "  9 CORRESPONDS TO A D
     IGIT TO PLACE IN": PRINT "TH
     AT POSITION OF THE MASK": PRINT
     "  0 CORRESPONDS TO A NONZER
     O DIGIT TO"
 60  PRINT "PLACE IN THAT POSITION
     . IF YOU WANT A": PRINT "COM
     MA OR COLON IN THE MASK, ENC
     LOSE THE"
 65  PRINT "ENTIRE MASK IN QUOTES
     TO INPUT IT.": PRINT
 70  INPUT "EDIT MASK? ";MASK$
 75  INPUT "NUMBER TO EDIT? ";NUM:
      GOSUB 100: PRINT "EDITED NU
     MBER:";OUT$
 80  PRINT : INPUT "1=NEW NUMBER,
     2=NEW MASK AND NUMBER?";N
 85  ON N GOTO 75,70
 90  GOTO 80
100  NUM$ =  STR$ (NUM):LN =  LEN
     (NUM$):LM =  LEN (MASK$):QM =
     0:QN = 0:RM = 0:RN = 0:PN =
     0:PM = 0:NF% = 0:MF% = 0:FD%
      = 0:DF% = 0
110  OUT$ = "": IF NUM < 0 THEN NF
     % = 1: REM  SET FLAG TELLING
      WHETHER INPUT NUMBER IS NEG
     ATIVE
120  IF  RIGHT$ (MASK$,1) = "-" THEN
     MF% = 1: REM  SET FLAG TELLI
```

```
      NG WHETHER INPUT MASK HAS TR
      AILING MINUS SIGN
 130  IF  LEFT$ (MASK$,1) = "$" THEN
      FDZ = 1: REM  SET FLAG TELLI
      NG WHETHER INPUT MASK HAS FL
       OATING DOLLAR SIGN
 140  FOR I = 1 TO LM: REM  FIND P
      OSITION OF DECIMAL POINT IN
      MASK
 150  IF  MID$ (MASK$,I,1) = "." THEN
      PM = I
 160  NEXT : IF FDZ = 0 THEN DFZ =
      1: REM  IF NO FLOATING DOLLA
      R SIGN IN MASK, SET FLAG SAY
      ING "$" ALREADY OUTPUT TO ED
      ITED FIELD
 170  FOR I = 1 TO LN: REM  FIND P
      OSITION OF DECIMAL POINT IN
      NUMBER TO EDIT
 180  IF  MID$ (NUM$,I,1) = "." THEN
      PN = I
 190  NEXT
 200  IF PN THEN RN = LN - PN: REM
       IF DECIMAL POINT IN NUMBER,
       COMPUTE # DIGITS RIGHT OF D
      ECIMAL PT.
 210  IF PM = 0 THEN 250: REM  IF
      DEC. PT. IN MASK, FIND # DIG
      IT POSITIONS RIGHT OF IT
 220  FOR I = LM TO PM STEP  - 1
 230  IF  MID$ (MASK$,I,1) = "0" OR
       MID$ (MASK$,I,1) = "9" THEN
      RM = RM + 1
 240  NEXT
 250  IF PN = 0 AND PM = 0 THEN 30
      0
 260  IF RM = RN THEN 300
 270  IF RM < RN THEN 290
 280  FOR I = RN TO RM - 1:NUM$ =
      NUM$ + "0": NEXT : GOTO 300:
       REM  ZERO-FILL RIGHTMOST DE
      CIMAL POSITIONS OF NUM$
 290  I = LN - RN + RM - 1:NUM$ =  LEFT$
      (NUM$,I) + "*": REM  TRUNCAT
      E NUM$ TO MATCH MASK, PUT "*
      " IN RIGHTMOST DIGIT
 300  QN =  LEN (NUM$) - RM: IF PN THEN
      QN = QN - 1: REM  GET # DIGI
      TS LEFT OF DEC. PT. IN NUMBE
      R, IGNORING DEC. PT., IF ANY

 310  IF NFZ AND MFZ THEN QN = QN -
      1: REM  IGNORE MINUS SIGN IN
       NUMBER IF TRAILING MINUS IN
       MASK
 320  FOR I = 1 TO LM: IF I = PM THEN
      350: REM  FIND # DIGITS IN M
      ASK LEFT OF DEC. PT.
 330  IF  MID$ (MASK$,I,1) = "0" OR
       MID$ (MASK$,I,1) = "9" THEN
      QM = QM + 1
 340  NEXT
 350  IF QM >  = QN THEN 370: REM
       TRUNCATE NUMBER ON LEFT, MA
       KING LEFTMOST DIGIT "*"
 360  I =  LEN (NUM$) - QN + QM - 1
       : IF NFZ AND MFZ THEN I = I -
      1: REM  DROP MINUS SIGN ALSO
       IF IGNORED BEFORE
 365  NUM$ = "*" +  RIGHT$ (NUM$,I)
      :QN = QM
 370  I1 = 1: IF FDZ THEN I1 = 2: REM
       WILL IGNORE ANY FLOATING DO
      LLAR SIGN IN MASK
 380  I2 = LM: IF MFZ THEN I2 = LM -
      1: REM  WILL IGNORE ANY TRAI
      LING MINUS IN MASK
 385  IF NFZ AND MFZ AND  LEFT$ (N
      UM$,1) = "-" THEN QN = QN +
      1: REM  IF NUMBER'S MINUS SI
      GN WAS IGNORED BEFORE, PUT I
      T BACK IN
 390  IF PN THEN NUM$ =  LEFT$ (NU
      M$,QN) +  RIGHT$ (NUM$,RM): REM
       DROP DEC. PT. FROM NUMBER S
      TRING
 400  IF NFZ AND MFZ AND  LEFT$ (N
      UM$,1) = "-" THEN NUM$ =  RIGHT$
      (NUM$, LEN (NUM$) - 1): REM
       DROP MINUS SIGN IF TRAILING
       MINUS IN MASK
 410  J =  LEN (NUM$): FOR I = I2 TO
       I1 STEP  - 1:M$ =  MID$ (MAS
      K$,I,1):N$ = " ": IF J > 0 THEN
       N$ =  MID$ (NUM$,J,1)
 420  IF M$ < > "," THEN 490
 430  IF N$ < > "-" THEN 450
 440  OUT$ = N$ + OUT$:J = J - 1: GOTO
       550
 450  IF N$ < > " " THEN 480
 460  IF DFZ THEN 440: REM  IF FLO
      ATING DOLLAR SIGN ALREADY OU
      TPUT, GO INSERT BLANK
 470  DFZ = 1:OUT$ = "$" + OUT$: GOTO
       550
 480  OUT$ = M$ + OUT$: GOTO 550
 490  IF M$ < > "9" THEN 520
 500  IF N$ = " " THEN 460: REM  I
      F ALL DIGITS OF NUMBER OUTPU
      T, GO OUTPUT FLOATING DOLLAR
```

```
      SIGN OR BLANK
510   GOTO 440: REM  GO OUTPUT THE
      DIGIT
520   IF M$ < > "0" THEN 480: REM
      GO OUTPUT CURRENT CHARACTER
      IN MASK
530   IF N$ < > "0" THEN 500: REM
      GO OUTPUT BLANK OR DIGIT
540 N$ = " ": GOTO 440: REM  OUTP
    UT BLANK
550   NEXT : IF DF% = 0 THEN OUT$ =
      "$" + OUT$: REM  IF FLOATING
      DOLLAR NOT OUTPUT, APPEND I
    T ON LEFT
555   IF DF% AND FD% THEN OUT$ = "
      " + OUT$: REM  IF DOLLAR SI
    GN ALREADY OUTPUT, PUT BLANK
      IN PLACE OF MASK'S DOLLAR S
    IGN
560   IF MF% = 0 THEN  RETURN : REM
      ALL DONE IF NO TRAILING MIN
    US IN MASK
570 N$ = " ": IF NF% THEN N$ = "-
      ": REM  BLANK IF POSITIVE, M
    INUS SIGN IF NEGATIVE
580 OUT$ = OUT$ + N$: RETURN
```


## STOCK MARKET ANALYSIS PROGRAM
## DJI WEEKLY AVERAGE 1897-DATE

ANA1* (ANALYSIS 1) is a set of BASIC Programs which enables the user to perform analyses on the Dow Jones Industrial weekly average data. From 6 months to 5 years of user selected DJI data can be plotted on the entire screen in one of 5 colors using Apples' High Resolution capabilities. The DJI data can be transformed into different colored graphic representations called transforms. They are: user specified moving averages; a least squares linear fit (best straight line); filters for time, magnitude, or percentage changes; and user created relationships between the DJI data, a transform, or a constant using +,-,x,/ operators. Colored lines can be drawn between graphic points. Graphic data values or their dates of occurrence can be displayed in text on the screen. Any graph or text can be outputted to a users printer. The Grid Scale is automatically set to the range of the graphs or can be user changed. As many colored graphs as wanted can be plotted on the screen and cleared at any time. The user can code routines to operate on the DJI/transform data or create his own disk file data base. ANA1 commands can be used with his routines or data base. An Update program allows the user to easily update the DJI file with current DJI weekly data.

The ANA1 two letter user commands are: CA = Calculate, no graph. CG = Clear Graphs, leave Grids. CK = Checking out program, known data. CO = Color of next graph (red, green, violet, white, blue). CS = Clear Screen. DL = Draw Line between points. FI = Filter data for time, magnitude, or percent change. FU = Data, transform, or constant Function with +,-,x,/ operator. GD = Graphic mode, display all Graph Data on screen. GR = Graph data to screen. GS = Set Grid Scale. HE = Help, summary of any commands usage. LD = Load Data from disk file from inputted date to memory. LG = Leave Graphs, automatic Grid rescaling. LO = Look, select a range of the LD data and GR; All commands can now be used on this range. LS = Least squares linear fit of the data. MA = Moving Average of the data. NS = No Scale, next graph on screen does not use Grid Scale. NT = No Trace. PR = User implimented Printer routine. TD = Text mode, display Text Data on screen. TI = Time number to date or vice versa. TR = Trace. TS = Text Stop for number of lines outputted to screen when in TD. U1/U2 = User 1/2 implimented routines. VD = Values of Data outputted in text. VG = Values of Grid; low/high/delta. VT = Values of Transform outputted in text.

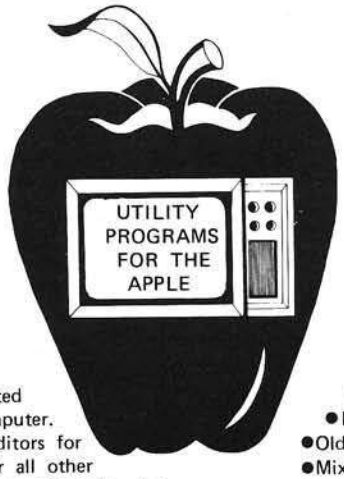
**APPLE® II, 48 K, APPLESOFT ROM CARD, DISK II DOS 3.2**
**ANA1 DISK & MANUAL . . . $49.95**
**(CA residents add 6% sales tax)**

**GALAXY**
**DEPT. A02**
**P.O. BOX 22072**
**SAN DIEGO, CA 92122**

* Software Review in Call-A.P.P.L.E. (2/80): "An example of an excellent piece of software exploiting most of Apple II's major features." Overall Rating = 92.1

* Software Review in Apple Orchard (3/80): "A remarkably flexible approach to the analysis and plotting of any time series data." Overall Rating = 85.7

# SYNERGISTIC SOFTWARE




## *Program Line Editor*

by Neil Konzen

Stronger than the Integer
More powerful than Applesoft
Faster than a speeding cursor
Able to leap the longest statements in a single move.

The Program Line Editor is the most sophisticated program development tool for the Apple II computer. Program Line Editor includes full feature line editors for both Integer and Applesoft programs that render all other techniques obsolete. Program development and modification can now be performed in a tenth the time previously required.

- More than 11 editing commands such as insert, delete, find & pack.
- Versatile escape functions execute a complex user defined command with just 2 keystrokes.
- Complete listing control (stop list, resume, abort)
- Invisible and indestructible routines actually become part of DOS and are unaffected by INT, FP, or MAXFILES.

Available on disk only for any Apple][ or Apple][ Plus for $40.00. Introductory price of $37.50 thru April 1980.

## Higher Text

by Darrell & Ron Aldrich

When the Apple's text output isn't good enough, Higher Text gives you the flexibility of:

- Full 96 characters of ASCII upper and lower case.
- Colorful text in 10 high resolution colors.
- Foreign alphabets or scientific symbols.
- Old English, script, or even sideways text.
- Mixed text and graphics anywhere on the high resolution screen.
- Choice of normal 7 x 8 characters or detailed 14 x 16 dot characters.
- Regular, extra tall, extra wide, or large characters.

Higher Text allows you to design unique character sets with the easy to use character editors. Graphs, game boards, and program logos can be labeled with this versatile package. Now you can create professionally customized displays using routines that can be easily linked to any Applesoft or Integer BASIC program.

Available on disk only for any Apple ][ or Apple ][ Plus for $35.00. Introductory price of $32.50 thru April 1980.

Send check or inquiry to Synergistic Software, 5221 120th Ave. S.E., Bellevue, WA 98006
Washington residents add 5.3% sales tax.

# 6502
# MACRO ASSEMBLER

# AND TEXT EDITOR

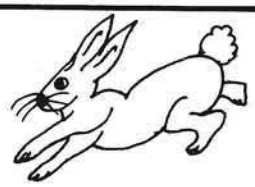
- Versions for PET, APPLE II, SYM, KIM and ATAR I (1st quarter 1980)
- Written entirely in **machine language**
- Occupies 8K of memory starting at $2000 — Apple version with disk occupies just over 9K
- **Macro** and conditional assembly
- 36 error codes, 26 commands, 22 pseudo ops
- Labels up to 10 characters
- Auto line numbering and renumber command
- String search and string search and replace
- Copy, move, delete, load, save, and append commands

**Cassette and Manual $49.95**
(including U. S. postage)

## Eastern House Software

**3239 Linda Dr.** **Winston-Salem, N. C. 27106**

# PET
# RABBIT



Load, Save, Verify, Execute
8 K in 38 seconds versus
PETs 2 Minute 45 seconds,
**plus more!**

High-speed Cassette Routines work with 8K, 16K, or 32K **new ROM PETs** which have the new Commodore cassette deck (like the external version which sells for $95.00). Note: If you have a new ROM PET with the old style lift-top deck, everything but the high-speed cassette routines will work.

— **Auto repeat** of any key held down, toggle character set.
— **RAM Memory Test**, convert #'s to hex and decimal.

**12** Rabbit Commands

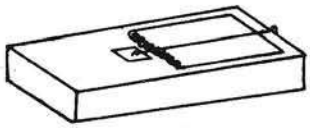
Note: Rabbit is 2K of machine code at $1800 for 8K PETS, $3000 or $3800 for 16K PETS, or $7000 or $7800 for 32K PETS. (Specify one of the 5 versions.)

**Cassette and Manual — $29.95** (Add $5.00 for foreign)

## Eastern House Software

**3239 Linda Dr.** **Winston-Salem, N. C. 27106**

# TRAP
# 65



For PET, APPLE, SYM, OSI, etc.
Any 1 mhz 6502!

— Prevents from "hanging up" via execution of **unimplemented opcodes!** Causes your monitor to **display location** of bad op code!
— Our computers have stopped crashing!
— **Debugging easier!**
— **Easy to install** — plugs into 6502's socket.

TRAP 65 is currently being used to insure that there are no bad opcodes in programs before submittal for ROM masks!

We are using TRAP 65 in experiments on extending the 6502's instruction set — PHX, PHY, INCD.

**$149.95** (Add $4.00 postage, $10.00 foreign)

## Eastern House Software

**3239 Linda Dr.** **Winston-Salem, N. C. 27106**

# Finally, MAE — A PET DISK-Based MACRO ASSEMBLER/TEXT EDITOR

Works with 32K PET

— Works with 2040 Disk, and can drive 2022/2023 Printer, and/or RS232/20 ma Device thru User Port.
— 100% Disk Based, 100% Machine Language.
— Macros, Conditional Assembly, and a new feature we developed called Interactive Assembly.
— Coexists with Basic, Auto character repeat, Sorted Symbol Table.
— 27 Commands, 26 Pseudo Ops, 5 Conditional Ops, 38 Error Codes.
— Creates relocatable object code on disk.
— Assemble from Memory or Disk.
— String search, search and replace, and inter-line edit.
— Auto line #-ing, move, copy, delete, renumber.
— Labels up to 31 characters — user specifies length.
— Includes extention to PET monitor (disassemble, trace, etc.), Library of PET ROM locations, Relocating Loader, plus more.

**Manual, Diskette, U. S. postage — $169.95**
(Requires completion of License Agreement — Write for details)

## Eastern House Software

**3239 Linda Drive** **Winston-Salem, N. C. 27106**

# PET Keysort Update

**Two changes are presented to improve the PET Keysort.**

**Rev. James Strasma**
**120 West King Street**
**Decatur, IL 62521**

After further use and testing, I decided to make two changes to my program KEYSORT as printed in MICRO:23. First, I've added to the intelligence of the himem setter in lines 550-630 of the source listing. Previously, my copy wasted about 100 bytes of memory by setting himem lower than it needs to be. Now it is set just at the start of the sort. The new source listing would read:

```
.550      lda *him + 1;cut himem?
.560      seo
.570      sbo #h,sart
.580      boo sav      dont lower himem
.590      bne cut      ;do lower it
.600      lda *him     depend on lo byte
.605      sbo #L,sart
.610      boo sav      ;if already lower
.615cut   lda #L,sart  ;out lo, then hi
.620      sta *him
.625      lda #h,sart
.630      sta *him + 1
```

This is an addition of 3 lines and 5 bytes.

```
.: 7C50 00 00 00 00 A5 35 38 E9
.: 7C58 7C 90 10 D0 06 A5 34 E9
.: 7C60 54 90 08 A9 54 85 34 A9
.: 7C68 7C 85 35 20 98 7D A5 2C
.: 7C70 85 15 A5 2D 85 16 A5 02
.: 7C78 C9 24 F0 06 A9 80 85 00
.: 7C80 85 01 A0 00 B1 15 C5 00
.: 7C88 F0 06 A9 80 C5 00 D0 08
.: 7C90 C8 B1 15 C5 01 30 01 C8
.: 7C98 98 AA A5 03 C9 25 F0 04
.: 7CA0 A9 09 85 04 A5 05 C9 23
.: 7CA8 F0 04 A9 00 85 06 A0 02
.: 7CB0 B1 15 18 65 15 85 18 C8
.: 7CB8 B1 15 65 16 85 19 A5 2F
.: 7CC0 C5 19 F0 02 B0 0E A5 2E
.: 7CC8 C5 18 F0 02 B0 06 E0 02
.: 7CD0 B0 11 90 17 E0 02 B0 0B
.: 7CD8 A5 18 85 15 A5 19 85 16
.: 7CE0 18 90 9F A0 04 B1 15 C9
.: 7CE8 01 F0 08 20 98 7D A2 80
```

The other change is in the way KEYSORT handles nulls. Logically, they should have a value below any other character. The original KEYSORT treats them this way. However, that leads to a problem with partially filled Basic arrays. All the undefined array elements start out as nulls, and end up after a sort at the 'bottom' of the array, where the significant elements were before. I elected to redefine nulls as larger than 'Z', so they stay at the 'top' of the array, where they were before the sort. The necessary changes are made in lines 5180-5220 of the source listing. The label 'null' is deleted from line 5180. A new label, 'same' is added to line 5220. Then 4 new lines are added after line 5420. These are like lines 5180-5200, except that the destinations are opposite. The new lines read:

```
.5422nullcpx#1
                    ;put nulls @top of$s
.5424     beq two((
                    ;to keep them out
.5426     bpL one((
                    ;of the way of prgm
.5428     bmi same
                    ;jump
```

This change adds 4 lines and 8 bytes to the program. Unfortunately it also alters many other parts of the object code in order to stay just below himem, so you'll need to check the enclosed new object listing carefully against your copy of the former version.

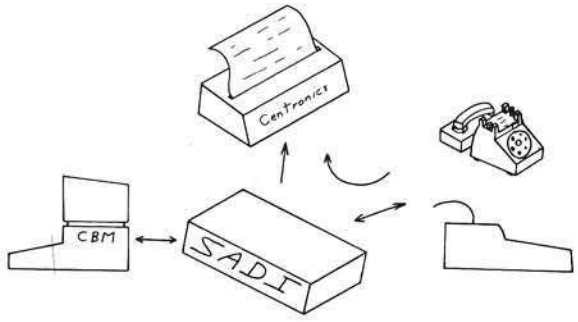
After these changes are made, the system call address for the sort is lowered, to sys (31828). The option setting addresses are unchanged. If you'd rather not make these changes yourself, updated copies for any loation in memory, or for old ROMs are available directly from the author for $5.00. Please specify the address and ROM set you prefer to use.

```
.: 7CF0 4C 57 C3 A0 06 B1 15 85
.: 7CF8 12 88 B1 15 85 13 18 6A
.: 7D00 85 0E A5 12 6A 18 69 01
.: 7D08 85 0D A5 0E 69 00 85 0E
.: 7D10 A5 15 18 69 04 85 23 A5
.: 7D18 16 69 00 85 24 A5 12 85
.: 7D20 0B A5 13 85 0C A5 0E F0
.: 7D28 03 4C AA 7D A5 0D C9 01
.: 7D30 D0 F7 A5 0B 85 1B A5 0C
.: 7D38 85 1C 20 B3 7F A0 00 B1
.: 7D40 15 85 20 C8 B1 15 85 21
.: 7D48 C8 B1 15 85 22 A5 23 18
.: 7D50 69 03 85 18 A5 24 69 00
.: 7D58 85 19 B1 18 91 15 88 B1
.: 7D60 18 91 15 88 B1 18 91 15
.: 7D68 38 A5 0B E9 01 85 0B A5
.: 7D70 0C E9 00 85 0C C9 00 D0
.: 7D78 57 A5 0B D0 53 A5 07 85
.: 7D80 1B A5 08 85 1C 20 B3 7F
.: 7D88 A5 20 A0 00 91 15 C8 A5
```

```
. 7D90 21 91 15 C8 A5 22 91 15
. 7D98 A2 24 BD DB 7F 48 B5 00
. 7DA0 9D DB 7F 68 95 00 CA 10
. 7DA8 F1 60 38 A5 0D E9 01 85
. 7DB0 0D A5 0E E9 00 85 0E 85
. 7DB8 1C A5 0D 85 1B 20 B3 7F
. 7DC0 A0 00 B1 15 85 20 C8 B1
. 7DC8 15 85 21 C8 B1 15 85 22
. 7DD0 A5 0D 85 09 A5 0E 85 0A
. 7DD8 A5 09 85 07 A5 0A 85 08
. 7DE0 18 26 09 26 0A A5 0A C5
. 7DE8 0C F0 05 90 0D 4C 88 7E
. 7DF0 A5 09 C5 0B 90 04 F0 5E
. 7DF8 B0 F3 A5 09 85 1B A5 0A
. 7E00 85 1C 20 B3 7F A0 00 B1
. 7E08 15 85 17 C8 B1 15 85 18
. 7E10 C8 B1 15 85 19 18 A5 15
. 7E18 69 03 85 15 A5 16 69 00
. 7E20 85 16 A0 02 B1 15 85 1C
. 7E28 88 B1 15 85 1B 88 B1 15
. 7E30 85 1A A5 19 85 1F A5 18
. 7E38 85 1E A5 17 85 1D 20 D7
. 7E40 7E A5 14 C5 17 90 0F F0
. 7E48 0D 18 A5 09 69 01 85 09
. 7E50 A5 0A 69 00 85 0A A5 09
. 7E58 85 1B A5 0A 85 1C 20 B3
. 7E60 7F A0 02 B1 15 85 1C 88
. 7E68 B1 15 85 1B 88 B1 15 85
. 7E70 1A A5 22 85 1F A5 21 85
. 7E78 1E A5 20 85 1D 20 D7 7E
. 7E80 A5 14 C5 17 F0 02 B0 1E
. 7E88 A5 07 85 1B A5 08 85 1C
. 7E90 20 B3 7F A0 00 A5 20 91
. 7E98 15 C8 A5 21 91 15 C8 A5
. 7EA0 22 91 15 4C 25 7D A5 07
. 7EA8 85 1B A5 08 85 1C 20 B3
. 7EB0 7F A5 15 85 18 A5 16 85
```

```
. 7F90 17 60 B1 1E 85 14 A9 00
. 7F98 85 17 60 A9 00 85 14 B1
. 7FA0 1B 85 17 60 85 14 B1 1E
. 7FA8 85 17 60 E0 01 F0 EC 10
. 7FB0 E1 30 D8 A5 1B 85 1E A5
. 7FB8 1C 85 1F 18 26 1B 26 1C
. 7FC0 A5 1B 18 65 1E 85 1B A5
. 7FC8 1C 65 1F 85 1C A5 1B 18
. 7FD0 65 23 85 15 A5 1C 65 24
. 7FD8 85 16 60 00 00 00 00 00
. 7EB8 19 A5 09 85 1B A5 0A 85
. 7EC0 1C 20 B3 7F A0 00 B1 15
. 7EC8 91 18 C8 B1 15 91 18 C8
. 7ED0 B1 15 91 18 4C D8 7D A0
. 7ED8 00 84 0F A6 06 D0 0B A5
. 7EE0 1D 85 10 A5 1A 85 11 18
. 7EE8 90 70 A5 04 D1 1E F0 08
. 7EF0 C8 C4 1D 90 F7 4C EB 7C
. 7EF8 84 0F CA F0 02 B0 F1 C8
. 7F00 D1 1E F0 05 C4 1D 90 F7
. 7F08 C8 88 98 38 E5 0F 85 10
. 7F10 E6 0F A5 0F 18 65 1E 85
. 7F18 1E A5 1F 69 00 85 1F A0
. 7F20 00 84 0F A6 06 A5 04 D1
. 7F28 1B F0 08 C8 C4 1A 90 F7
. 7F30 4C EB 7C 84 0F CA F0 02
. 7F38 B0 F1 C8 D1 1B F0 05 C4
. 7F40 1A 90 F7 C8 88 98 38 E5
. 7F48 0F 85 11 E6 0F A5 0F 18
. 7F50 65 1B 85 1B A5 1C 69 00
. 7F58 85 1C A5 10 C5 11 F0 08
. 7F60 B0 0C 85 0F A2 01 D0 0C
. 7F68 85 0F A2 00 F0 06 A5 11
. 7F70 85 0F A2 02 C9 00 F0 33
. 7F78 A0 00 B1 1B D1 1E D0 24
. 7F80 C8 C4 0F 90 F5 E0 01 F0
. 7F88 09 10 10 A9 00 85 14 85
```


# PET TWO-WAY RS-232 and PARALLEL OUTPUT INTERFACE




SADI - The microprocessor based serial and parallel interface for the Commodore PET. SADI allows you to connect your PET to parallel and serial printers, CRT's, modems, acoustic couplers, hard copy terminals and other computers. The serial and parallel ports are independent allowing the PET to communicate with both peripheral devices simultaneously or one at a time. In addition, the RS-232 device can communicate with the parallel device.

Special Features for the PET interface include:
- Conversion to true ASCII both in and out
- Cursor controls and function characters specially printed
- Selectable reversal of upper and lower case
- PET IEEE connector for daisy chaining
- Addressable - works with other devices

Special Features for the serial interface include:
- Baud rate selectable from 75 to 9600
- Half or full duplex
- 32 character buffer
- X-ON, X-OFF automatically sent
- Selectable carriage return delay

Special Features for the parallel interface include:
- Data strobe - either polarity
- Device ready - either polarity
- Centronics compatible

Complete with power supply, PET IEEE cable, RS-232 connector, parallel port connector and case. Assembled and tested.

SADIa (110VAC) $295
SADIe (230VAC) $325

Order direct or contact your local computer store.

**CONNECTICUT microCOMPUTER, Inc.**
*150 POCONO ROAD*
*BROOKFIELD, CONNECTICUT 06804*
*TEL: (203) 775-9659 TWX: 710-456-0052*

VISA AND M/C ACCEPTED - SEND ACCOUNT NUMBER, EXPIRATION DATE AND SIGN ORDER.
ADD $3 PER ORDER FOR SHIPPING & HANDLING - FOREIGN ORDERS ADD 10% FOR AIR POSTAGE.

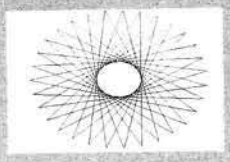
# Low Cost Graphics Are Now Here

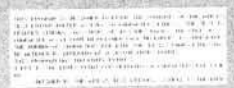
## Add Printable High Resolution (320 x 200) Graphics To Your AIM-65 Microcomputer

**Visible Memory Board** It's a reality. The Visible Memory and graphic print software from MTU are now available for the Rockwell AIM-65. The Visible Memory gives a high resolution 320 wide by 200 high bit mapped pixel display matrix. Each dot is individually addressable for maximum utilization and speed. Thus characters, image shapes, and graphs can all be displayed separately or simultaneously if needed — maximum flexibility for you.

The Visible Memory is just that, an 8K byte RAM board that contains 2 access ports to the memory matrix. The microprocessor bus uses one port and the display refresh circuitry uses the second port. The contents of the memory bit-for-bit is precisely what is displayed. If you need 8K of RAM for a non-display application, use it! It makes no difference to the board what its contents are; program (seen in its binary pattern form) or a human recognizable display pattern. The display refresh occurs at times when the processor never goes to memory. Therefore there is no snow on the display and no wait states for the processor.

**Hardcopy Too** In addition MTU has engineered a software package to drive the AIM-65 printer in new ways. Three new forms of printing are possible. QUICKPRINT gives a matrix 200 across by 320 up the 2¼" wide paper. QUALITYPRINT gives two prints, each 100 x 320 which gives a higher quality (4½" wide) printed area when placed side by side. TEXTPRINT allows you to print the AIM text buffer area of memory as 10 rows of characters printed "up" the paper strip. You may specify up to 127 characters per row for the row length. The QUICK and QUALITY print modes are designed to give you fast, easy hardcopy of the Visible Memory contents. Thus you now have a graphic computer with hardcopy capability.





**Graphic/Text Software Drivers** To allow you to easily use this graphic display and print power, MTU has also designed the K-1008-5C software package which gives you point plotting, line drawing, character generation and a host of other subroutines. Written in assembly language, these routines may be executed from BASIC or assembly language — your choice. Text output from BASIC or the AIM monitor may also be shown on the Visible Memory display as up to 22 lines by 53 characters per line significantly enhacing the use of the AIM-65 as a computer with a CRT display.

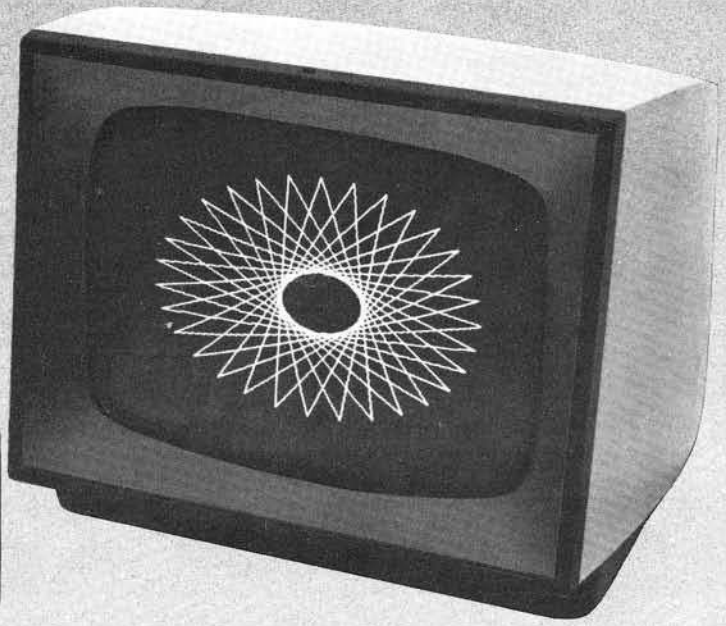
**Call Us Now** Many educators have been waiting for this type of price/performance to set up courses. MTU will be pleased to quote quantity purchases — call us direct — now. Demand is high and Fall is just around the corner.

| | |
|---|---|
| K-1008 Visible Memory | $240 |
| K-1008-5C Graphic/text software | $25 |
| K-1009-1C AIM Print software | $25 |
| K-1000-5 AIM-65 Power Supply | $65 |
| K-1005-A AIM-65 Card File | $85 |
| Many others not listed | |

Call or write for our catalog listing all our AIM-65 (also our PET, KIM-1, and SYM-1) products.

As of June 1, 1980 place orders at: Micro Technology Unlimited, P.O. Box 12106, 2806 Hillsborough Street, Raleigh, North Carolina 27605

MTU
**Micro Technology Unlimited**
P.O. Box 4596, 841 Galaxy Way
Manchester, N.H. 03108
603-627-1464



# NOW YOUR APPLE II CAN PERFORM JUST LIKE THE BIG BOYS

If you're a businessman who demands ultimate performance from your Apple II, then take a look at this outstanding General Ledger Package from Small Business Computer Systems (SBCS).

**It features**

- 6 digit account numbers
- 31 character account name.
- Ten levels of subtotals — giving you a more detailed income statement and balance sheet.
- Departmentalizing . . . up to nine departments.
- Flexibility — adaptable to any printer and either cash or accural accounting methods.
- Cash Journal allows a 33 character transaction description and automatically generates the appropriate offsetting entry.
- You can print the balance sheet and income statement for the current month, current quarter, or any of the previous three quarters. This year's or last year's totals are also included on the income statement. Or a special report that lists the current account balance for selected accounts.
- Higher number of entries from an external source — as many as 1,000 per session.
- No limit on entries — giving you the opportunity to make your entries as many times or as often as you want.
- With high speed printer routines and other special features of our conversion, processing performance does not decrease dramatically at the system limits.
- Look at these examples of times required to update the chart and print the audit trail.
  With 133 item chart of accounts, 700 postings into 70 regular accounts: less than 20 min.
  With 133 item chart of accounts, 1000 postings into 70 regular accounts: less than 30 min.
  With 210 item chart of accounts, 1000 postings into 125 regular accounts: less than 40 min.
- Coming early this year — capability to archive up to 2,500 postings. The chart of accounts will also be archived to maintain the opening balance for the archive period.

**In the final analysis, your financial statements are what this General Ledger is all about. And with this General Ledger Package you can format your own balance sheet and income statement. As well, department financial statements may be formated differently. You have complete freedom to place titles and headings where you want them, skip lines or pages between accounts and generate subtotals and totals throughout the reports — up to ten levels if you need them.**

And coming early in 1980, SBCS will present the Accounts Payable/Accounts Receivable Package you have been waiting for.

Just compare these numbers against any package on the market today:*

| | 5 inch disc | 8 inch disc |
|---|---|---|
| Vendors or customers | 700 | 1,800 |
| Payable Transactions | 350 | 750 |
| Payable Invoices | 380 | 840 |
| Receivable Transactions | 600 | 1,300 |
| Receivable Invoices | 600 | 1,300 |

* These are maximum numbers that you can put on a disc if you're using the disc only for these respective data files.

**We are an authorized converter for Osborne/McGraw-Hill, providing you with business packages that will do everything the Osborne General Ledger will do in addition to many features we have added.**

*Call or write:*

**Small Business Computer Systems**
4140 Greenwood
Lincoln, Nebraska 68504
(402) 467-1878

# Software for APPLE® and OSI®
## by AMERICAN COMPUTER ENTERPRISES

## EXCLUSIVE GAMES for

### APPLE

**ROAD RALLYE 48K $14.95**
A 48K APPLE II game using 5 hires screens of increasingly harder courses that you must try to negotiate. Not just another game but a real challenge!

**UNCOPY 32K $29.95**
Now there is a way to make your disks uncopyable by altering APPLE D.O.S. Track and Sector configuration as well as changing the catalog track.

**The ULTIMATE TRANSFER 48K $25.00**
Easily upload or download APPLE programs hundreds or thousands of miles away over the phone. Easily transfer Integer, A-Soft, or Machine Language programs using your DC Hayes Micromodem.™

### OSI

***Game Pack 1***

**Featuring full use of the OSI color and sound with 3 game programs.**

**1) Horse Race** - Bet Win, Place or Show. Odds and horses are continually changing.
**2) Crash!** - Force your opponent to crash into the wall before you do.
**3)Airplane Invaders** - Similar to the popular arcade game using airplanes and bombs.
**C4P Disk - $12.95**

**OS • ELLO (8K) $9.95**
Play that ever popular board game with your C1P.

**OS • I RACEWAY $6.95**
A graphic horse race with changing odds. Bet to Win, Place or Show.

## The ULTIMATE MAIL LIST! - M-LABEL*

**One of the most comprehensive mailer programs ever offered for OSI. This Program is everything you ever wished you had in label generating software! Available only through CCNJ!**

**M-LABEL uses OS65D**

| | |
|---|---|
| C4-2P & C4MF (24 K) | - $69. |
| 32 K Version | - 79. |
| C2-8P & C8P | - 89. |
| C8S - C28S - C20EM | - 89. |
| C3 Series | - 89. |

*Requires 2 Drives

### OSI

**OS • ICE CREAM (8K) $11.95**
An interesting business simulation for adults & children. Try to make a profit running your own ice cream stand while typical business problems occur.

**SLOT MACHINE $6.95**
Try your luck at the slot machine. It has real Atlantic City type payoffs!

**OS • IMON & KALEIDOSCOPE $8.95**
How long can you last without making a fatal mistake? Up to 25 players. Included is kaleidoscope with unlimited patterns to tickle your fancy.

**TANK DRIVER $8.95**
Pilot your own tank through an enemy maze and reach your own base. Each time you play the course will change.

### OSI

**BOMBER WORD $9.95**
A truly unique game of word guessing. Each wrong letter will dispatch a bomber that will drop a bomb on a home. As the home burns, the inhabitants flee. Educational and fun too.

**MATH FUN $5.95**
A graphic board game that will not only entertain, but educate! Selectable skill levels.

**BARREL JUMPER $7.95**
See if you can jump over 15 barrels without falling or slipping on the ice. A lot harder than it looks!

**AIRPLANE INVADER $9.95**
Similar to that arcade game that is sweeping the country. Shoot down the rows of planes before the bomb falls on you.

## Order Your MICRO MUSIC BOARD
for the APPLE II from ***CCNJ*** and receive ***FREE 1 Digisong Pack*** of our choice. **$179.**

## COMPUTER CORNER of NEW JERSEY (201) 835-7080
439 Route 23, Pompton Plains, New Jersey 07444 Mastercharge and Visa Accepted!

# Expand KIM - 1 Versatility in Systems Applications

**Techniques and programs are presented which permit the simple addition of six sense switches or an ASCII keyboard to the KIM.**

**Ralph Tenny**
**P.O. Box 545**
**Richardson, TX 75080**

The KIM-1 microcomputer, produced by MOS Technology, Commodore and Rockwell International, is a single-board computer which gained early popularity with hobbyists. It also was adopted by industry for small controller applications. Some of these computers have been expanded into fairly large systems, in colleges as well as industry. One reason for the easy acceptance of the KIM-1 was the on-board keypad and six digit display. These features, along with a slow but extremely reliable audio cassette interface for program storage, made KIM-1 one of the first microcomputers which did not require an operator interface more expensive than itself.

The on-board keyboard and seven-segment display, which permits system operation without an teletype of terminal, is implemented in a way which permits addition of both an ASCII external keyboard and sense switches. Fig. 1 shows the key-pad implementation where U24 enables one of three banks of seven keys, and U2 (an MCS6530 programmable interface device detects a key closure in any one of the seven switch columns.

The keyboard encoding scheme works as follows: U2 is programmed for output on lines PB1-PB4 to dribe U24, a four line-to-ten-line decoder which has active-low outputs. Note that the least significant bit of U2's B port (PBO) is not used in the keyboard drive, so values written to Port B are incremented by two to select the next higher keybank. For example, writing 0016 to Port B selects Row keys, 0216 enables Row 1 and 0416 selects Row 2.

On Port A of U2 (lines PA0-PA6), which are programmed as inputs, a closure of (for example) key 8 will cause a logic zero to be input on PA5 whenever key Row 1 is active (low). KIM's operating system software then decodes Row 1/PA5 as key 8 and returns the value 0816 in the accumulator.

A fourth keybank (Row 3) is also implemented by this matrix, but the standard KIM-1 has only the TTY/KYBD switch installed on this row. FIG. 1 shows six additional switches implemented on Row 3; with proper programming, these can be used as sense switches or imput lines for address vectors in an expanded interrupt scheme. Listing 1 gives an example of the programming required to detect activity on Row 3 inputs.

The programming strategy required for any such inputs is to enable PAO-PA6 lines for input and sequentially activate the driving lines (outputs of U24 in this case) to their on (low) state. The program then reads all input lines, masks and inverts the data and returns to the calling program which tests the accumulator for any "one" bits. It is then the programmer's responsibility to repeat the scan periodically and test to see if the same data is present (a noise spike would be gone on a second scan) or has changed after some period of time. This testing allows for switch bounce—multiple closures of the contacts—a characteristic of all switches. Very good switches will bounce for a minimum of one or two milliseconds, while worn or cheap switches may bounce for up to 25 milliseconds. On the other hand, any operator who is trying to make a very short switch closure will find it difficult to release a switch earlier than 50 milliseconds after closure. Consequently, reading keys with software is a fine art!

LISTING I

```
A9 00        LDA #$00     SET PADD (KEY INPUT LINES)
8D 41 17     STA PADD     FOR INPUT
A9 3F        LDA #$3F     SET PBDD (ROW DEFINITION)
8D 43 17     STA PBDD     FOR OUTPUT
A9 06        LDA #$06     ENABLE KEYBOARD
8D 42 17     STA PBD      ON ROW 3
AD 40 17     LDA PAD      READ SENSE SWITCHES
29 7E        AND #$7E     MASK OFF TTY/KYBD SWITCH
49 7E        EOR #$7E     INVERT SWITCH DATA
A2 00        LDA #$00     DISABLE
8E 42 17     STX SBD      KEYBOARD
60           RTS          RETURN TO CALLING PROGRAM
```

Any keyboard with ASCII outputs is likely to have both a debounced output and a strobe which becomes active when there is a key pressed and the data has been debounced. Typically, the key data is active high (positive logic), but the strobe can be either active high or active low. The ASCII keyboard input described here does not use the strobe; instead, the key matrix is scanned in the same manner as is the normal KIM keypad. Fig. 2 shows the necessary connections—a pull-down transistor for each output bit of the keyboard. Any logic "one" data from the keyboard will input a low on the same lines as the KIM keypad. Note that some keyboards output only six bits, so the strobe can be implemented on Column G.

Listing 2 shows a "bare bones" scan program which will return to the calling program as did Listing 1. The basic scheme here is to initialize the accumulator to $FF_{16}$ and get the input data by a logic AND with the input port. The data is then inverted (Exclusive OR) and tested for any logic one bits. Note that the calling program could also permanently set the port for input and somewhat abbreviate the program segment shown. If the strobe is implemented on Column G as mentioned above, the 6502 BIT instruction followed by a test of the overflow status bit (BVC or BVS) will identify strobe activity. Note that the onboard keypad must not be active when the ASCII keyboard is being used, and that the normal KIM keypad scan routines will not properly interpret the ASCII input.

LISTING II

```
A9 80          LDA #$80      ENABLE KEYBOARD
8D 41 17       STA PADD      INPUT LINES
A9 FF          LDA #$FF      INITIALIZE ACCUMULATOR
2D 40 17       AND SAD       INPUT POSSIBLE KEYBOARD BITS
49 7F          EOR #$7F      INVERT ANY BITS PRESENT
F0 02          BEQ OUT       TEST FOR DATA PRESENT
A9 80          LDA #$80      SET FLAG FOR NO INPUT
60        OUT  RTS           RETURN TO CALLING PROGRAM
```

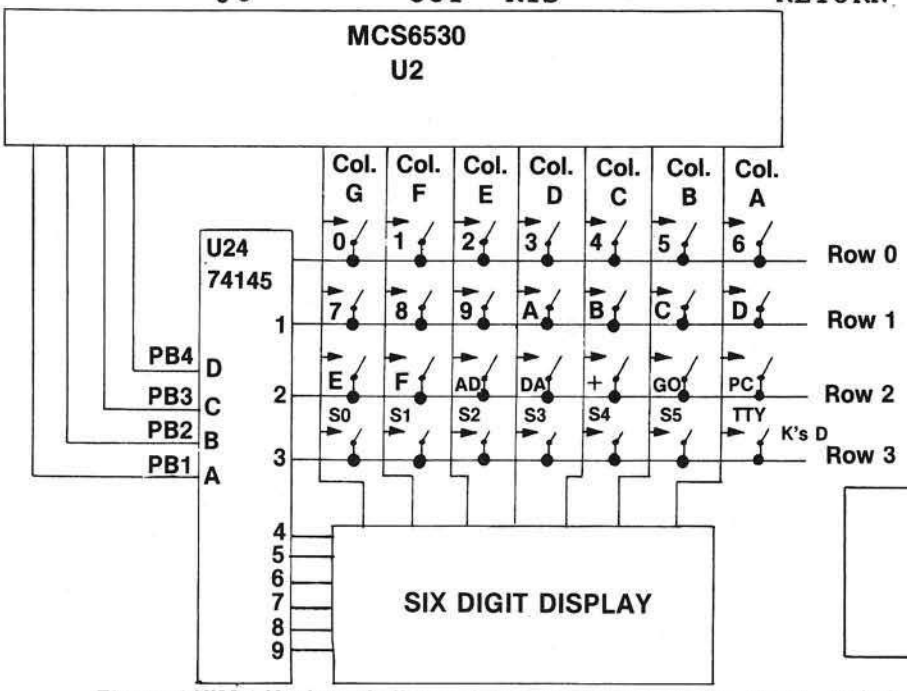



*Figure 1:* **KIM-1 Keyboard allows six sense switches to be added.**

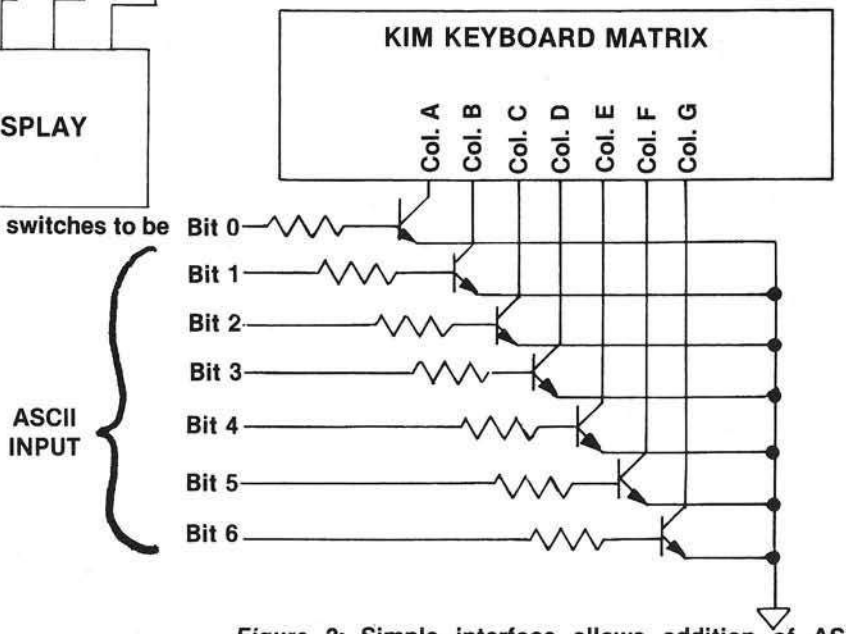



*Figure 2:* **Simple interface allows addition of ASCII keyboard to basic KIM-1.**

# *KIMSI FLOPPY DISKS—*

**PERRY PERIPHERALS HAS THE HDE MINIFLOPPY TO KIMSI ADAPTER**

**MINIFLOPPY S-100 ADAPTER: $15 + 1.50 p&h**
($20. as of June 1, 1980)

- **FODS and TED Diskette**
- **FODS and TED User Manuals**
- **Complete Construction Information**

(Not a kit; no parts supplied)

**OPTIONS:**

- **FODS Bootstrap in EPROM (1st Qtr'80)**
- **HDE Assembler (ASM) $75**
- **HDE Text Output Processor (TOPS) $135**

**(N.Y. State residents, add 7% Sales Tax)**

**Place your order with:**

**PERRY PERIPHERALS**
**P.O. Box 924**
**Miller Place, N.Y. 11764**
**(516) 744-6462**

Your "Long Island' HDE Distributor



**Decision Systems**

*Presenting the Other Side of the Apple II**

**INDEXED FILES**

**ISAM-DS** is an integrated set of routines for the creation and manipulation of indexed files. **ISAM-DS** provides capabilities comparable to those on large mainframes. You can rapidly retrieve records by key value or partial key value (retrieves any record in a 200 record file, 60 char/record, in less than 3 seconds compared to a maximum of 38 seconds for a **DOS** sequential file). Files never have to be reorganized. Duplicate key values may be used. Records may also be retrieved in sequence. **ISAM-DS** routines are easily integrated into Applesoft programs — they use less than **3K RAM** plus an index table.

Requires: Disk, Applesoft (32K ROM or 48K RAM)

$50

**STRUCTURED BASIC**

**PBASIC-DS** is a sophisticated preprocessor for structured **BASIC**. Now you can gain the power of **PASCAL**-like logic structures at a fraction of the cost. Use all regular **BASIC** statements plus 14 commands and 11 new statements/structures (**WHILE, UNTIL, CASE,** etc.). **PBASIC-DS** can be used to develop **INTEGER** or **APPLESOFT** programs. It is a great way to learn and use structured logic concepts.

Requires: Disk, Applesoft (32K ROM or 48K RAM)

$35

(Texas residents add 5% tax)

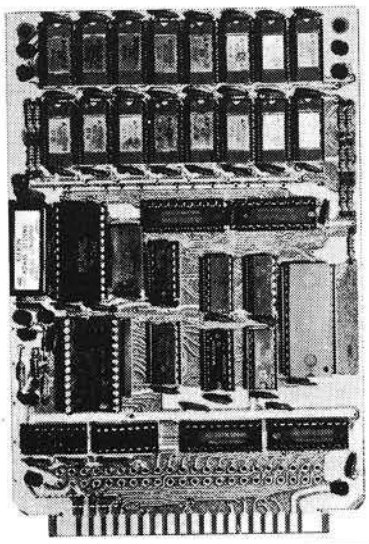
**Decision Systems**
**P.O. Box 13006**
**Denton, TX 76203**

*Apple II is a registered trademark of the Apple Computer Co.



## KIM/SYM/AIM-65—32K EXPANDABLE RAM

DYNAMIC RAM WITH ON BOARD TRANSPARANT REFRESH THAT IS COMPATIBLE WITH KIM/SYM/AIM-65 AND OTHER 6502 BASED MICROCOMPUTERS.



ASSEMBLED/TESTED
- WITH 32K RAM ..... $419.00
- WITH 16K RAM ..... $349.00
- WITHOUT RAM CHIPS ..... $279.00

HARD TO GET PARTS ONLY (NO RAM CHIPS) .. $109.00
BARE BOARD AND MANUAL ..... $49.00

- PLUG COMPATIBLE WITH KIM/SYM/AIM-65. MAY BE CONNECTED TO PET USING ADAPTOR CABLE. SS44-E BUS EDGE CONNECTOR.
- USES +5V ONLY (SUPPLIED FROM HOST COMPUTER BUS). 4 WATTS MAXIMUM.
- BOARD ADDRESSABLE IN 4K BYTE BLOCKS WHICH CAN BE INDEPENDENTLY PLACED ON 4K BYTE BOUNDARIES ANYWHERE IN A 64K BYTE ADDRESS SPACE.
- ASSEMBLED AND TESTED BOARDS ARE GUARANTEED FOR ONE YEAR, AND PURCHASE PRICE IS FULLY REFUNDABLE IF BOARD IS RETURNED UNDAMAGED WITHIN 14 DAYS.
- BUS BUFFERED WITH 1 LS TTL LOAD.
- 200NSEC 4116 RAMS.
- FULL DOCUMENTATION

**PET INTERFACE KIT $49.00**

CONNECTS THE ABOVE 32K EXPANDABLE RAM TO A 4K OR 8K PET. CONTAINS EXPANSION INTERFACE CABLE, BOARD STANDOFFS, POWER SUPPLY MODIFICATION KIT AND COMPLETE INSTRUCTIONS.

**6502, 64K BYTE RAM AND CONTROLLER SET**

MAKE 64K BYTE MEMORY FOR YOUR 6800 OR 6502. THIS CHIP SET INCLUDES:
- 32 M5K 4116-3 16KX1, 200 NSEC RAMS.
- 1 MC3480 MEMORY CONTROLLER.
- 1 MC3242A MEMORY ADDRESS MULTIPLEXER AND COUNTER.
- DATA AND APPLICATION SHEETS. PARTS TESTED AND GUARANTEED.

**$295.00 PER SET**

**16K X 1 DYNAMIC RAM**

THE MK4116-3 IS A 16,384 BIT HIGH SPEED NMOS, DYNAMIC RAM. THEY ARE EQUIVALENT TO THE MOSTEK, TEXAS INSTRUMENTS, OR MOTOROLA 4116-3.
- 200 NSEC ACCESS TIME, 375 NSEC CYCLE TIME.
- 16 PIN TTL COMPATIBLE.
- BURNED IN AND FULLY TESTED.
- PARTS REPLACEMENT GUARANTEED FOR ONE YEAR.

**$8.50 EACH IN QUANTITIES OF 8**

BETA COMPUTER DEVICES

1230 W. COLLINS AVE.
ORANGE, CA 92668
(714) 633-7280

CALIF RESIDENTS PLEASE ADD 6% SALES TAX
MASTERCHARGE & VISA ACCEPTED. PLEASE ALLOW 14 DAYS FOR CHECKS TO CLEAR BANK
PHONE ORDERS WELCOME

**ALL ASSEMBLED BOARDS AND MEMORY CHIPS CARRY A FULL ONE YEAR REPLACEMENT WARANTY.**

# MICRO Club Circuit

MICRO continues its soon to be monthly feature on 6502-related clubs. We are continuing to publish the names, locations, and activities of groups that could be of interest to our readers.

If you are involved in such a club have your representative register your group with us. In return for this registration we will send a free one year subscription of MICRO to your club's library. Include information regarding the club's name, location, algorithm, publications, purpose, officers, number of members, contact person, etc. Your club will then automatically appear in any club updates. If you are already registered please be sure to keep us current on your club's activities.

We would like this feature to be as helpful to our readers as possible. We welcome any information that will be of interest to other clubs: ie. what your club does, how it got started, what is published, your meeting format, purpose, etc.

We are publishing a complete list as of March. Please keep the updates coming! Start increasing your membership and give your group new exposure by telling others about yourselves.

If any of the following information is in error or outdated, please notify us.Address any questions or information to:

**MICRO CLUB CIRCUIT**
P.O. Box 6502
Chelmsford, MA 01824

**OSI User's Group**

Meets at Aristocraft on the first Thursday of the month (7-9:00 p.m.):
314 5th Avenue
New York, New York.
David Gillette, President. "Mutual aid and sharing of information."

**The Big Apple User's Group**

Meets on the last Tuesday of each month,at:
55-A Locust Avenue
New RochelleNew York
Tony Cerreta, President. "Exchange of ideas, growth in the field, production of hardware and software."

**Apple Pi Computer User's Group**

Meets first Thursday of each month at:
Colorado School of Mines
Cecil H. Green Bldg
Room 280
Boulder, CO.
Scott Knaster, President. "Spread information, use of documentation library and a software library for research and trading."

**Apple User's Group**

Meets on the third Thursday of each month (7:30 p.m.) at:
Computerland of Walnut Creek
1815 Ygnacio Valley Road
Walnut Creek, CA.
Hank Couden, President. "Foster knowledge and use of the Apple Computer."

**Original Apple Corps**

Meets second Sunday of the month (12:00 Noon) at:
Cal State University at Long Beach
Lecture Hall 151
Contact:
Kip J. Reiner,
19041-2 Hamlin Street
Reseda, CA 91335
"Expand the knowledge of Apple Computers. Software and Hardware."

**Greater Lafayette Apple User's Group**

Meets on the second Wednesday of each month (7:00 p.m.) at:
Digital Technology
10 North Third
Layfayette, IN. 47901
Jon W. Backstrom, President. "Library of public domain software. Exchange program. Want to educate members on successful programming skills. Workshops."

**Salem (Oregon) Area Computer Club**

Meets on the first Monday of each month. On odd numbered months, meetings are held at:
McKinley Community School
461 McGilchrist Street SE
Salem, OR.
and on even numbered months at:
The Computer Pathways Unlimited Retail Store
831 Lancaster Drive NE
South End-Lancaster Mall
Salem, OR.
Contact:
DougWalker
4554 Jan Ree Drive NE
Salem, OR 97303
"Each meeting features a presentation by a club member or an invited guest, followed by a 'bull session'."

**The Apple Cart**

A special interest group of American Mensa. For more information contact:
C. Brandon Gresham, Jr.
23 Van Buren Street
Dayton, OH 45402
"Hardware and software information. Software exchange. Promote the creation of well written and well documented software."

**Apple Sac**

Meets first Tuesday and third Wednesday of the month (7:00 p.m.) at:
Woodbridge School
5761 Brett Drive
North Highlands, CA
For further information contact:
Bill Norris, President
8074 Ruthwood Way
Orangevale, CA 95662
"To provide members with information, discounts at group rates, a place to exchange programs, ideas, techniques, hardware modifications and to feature guest speakers."

**Ohio Scientific Users NW**

Meets second Friday of each month at:
Data Systems Plaza
975 SE Sandy Blvd
Portland, OR 97214
Meetings are at 7:30 p.m.

**The Santa Barbara Apple User's Group**
Meets at:
2031 De La Vina
Santa Barbara, CA 93105

**Las Cruces Computer Club**
Meets the first Thursday of the month (7:30 p.m.) at:
SouthWest Computer Center
Suite 7
121 Wyatt Drive
Las Cruces, N.M. 88001
John Martellaro, President. Contact him at:
2929 Los Amigos, Apt.B
Las Cruces, New Mexico 88001

**Apple-Siders of Cincinnati**
Meets the second Tuesday of the month (7:30 p.m.) at:
The University of Cincinnati
Med. Science Bldg.
Contact:
John Anderson
5707 Chesapeake Way
Fairfield, OH 45014

**Denmark 6502 Club**
A country wide 6502 microprocessor club is being formed. Please contact for further information:
E.Skovgaard
Nordlundsvej 10
DK-2650 Hvidoure
Denmark
"Systems are reviewed and demonstrated. Developing a software library."

**Carolina Apple Core**
Meets third Tuesday (7:30 p.m.) of the month for general meeting. Other meetings are held on specific topics. Contact Joe Budge, President at:
P.O.Box 31424
Raleigh, NC 27622
"General support of the Apple User."

**North London Hobby Computer Club**
For more information contact:
Stephanie Bromley
The Polytechnic of North London
Holloway, London N7 8D8

**Computer Club in Belgium**
DeVlaamse Minicomputerclub
Lambrechtshoekenlaan 171b6
2060 Merksem, Belgium

**Apple Group - New Jersey**
Meets the fourth Friday of every month (7:00 p.m.) at:
Union County Tech. Institute
1776 Raritan Road
Scotch Plains, N.J.

**PACS PET User Group**
Meets the third Saturday (11:00 a.m.) every month at:
Science Building
LaSalle College
20th and Onley Avenue
Philadelphia, PA 19191

**Washington Apple Pi**
Meets the fourth Saturday (9:30 a.m.) every month at:
George Washington University
Rm. 206, Tompkins Hall
23rd and H streets NW
Washington, DC
You may write to this club at:
Washington Apple Pi
P.O.Box 34511
Washington, DC 20034
"Publishes a monthly newsletter."

**South Carolina Apple**
Meets second Tuesday of the month (7:30 p.m.) at:
The Byte Shop
1920 Blossom Street
Columbia, SC
You may address your inquiries to:
P.O.Box 70278
Charleston Heights, SC 29405

**WAKE—**
(Washington Area Kim Enthusiasts)
Meets the third Wednesday (7:30 p.m.) of every month at:
McGraw-Hill Continuing Education Center in Washington DC.
Contact Ted Beach at
5112 Williamsburg Boulevard
Arlington, VA 22207
for further information.

**Miami Apple User's Group**
Contact David Hall, Secretary at:
2300 NW 135th Street
Miami, FL 33167

**Sun Coast Apple Tree**
Meets the first and third Thursday of the month (7:00 p,m,) at:
The Computer Store
21 Clearwater Mall
Clearwater, FL 33516

**Central Ohio Apple Computer Hobbyists**
(COACH) Meets the third Saturday of each month (1:00 - 5:00). Contact:
Tom Mimlitch
1547 Cunard Road
Columbus, Ohio 43227

**Apple Dayton**
Meets the second Wednesday of odd numbered months and the second Thursday of even numbered months (7:30 p.m.) at:
Computer Solutions
Contact: Robert W. Rennard at
2281 Cobble Stone Court
Dayton, OH 45431

**Madison Pet User's Club**
Meets monthly at:
Washington Square Building
1400 East Washington Avenue
Madison, WI 53913
Contact: Ben A. Stewart
501 Willow
West Baraboo, WI 53913

**Micro and Personal Computer Club of St. Louis**
Meets monthly at:
Futureworld, Inc.
12304 Manchester Road
St. Louis, MO 63131
Contact: Mr. Kunihiro Tanaka

**Tulsa Computer Society**
Meets the last Tuesday of each month (7:30 p.m.) at:
Tulso Vo-Tech School
Seminar Center
3420 S. Memorial Drive
Tulsa, OK
This society also has an Apple User Group. For more information please write to:
The Tulsa Computer Society
P.O.Box 1133
Tulsa, OK 74101

**The Apple Corps**
Meets the second Saturday of each month (2-5:00 p.m.) at:
Greenhill School
14255 Midway Road
Dallas, TX

**Appleseed**
Meets monthly at:
The Computer Shop
6812 San Pedro
San Antonio, TX 78216

**Apples Brit.Columbia Computer Society**
Meets the first Wednesday of each month. Contact:
Gary B. Litte
101-2044 West Third Avenue
Vancouver, British Columbia
Canada V6J 1L5

**Honolulu Apple User's Society**
Meets the first Monday of each month at the Computerland Store in Honolulu. Contact:
Bill Mark
98-1451-A Kaahumanu Street
Aiea, Hawaii 96701

*Has anyone heard from the following clubs? Are they still active? Any current information would be appreciated!*

The MicroComputer Investor's Assoc.
The New England Apple Tree
Apple User Group of Europe

**Applelist**
Meets the second Wednesday of the month (7:30 p.m.) at:
Computerland
Skiff Street
Hamden, Conn.
Contact:
Marc B. Goldfarb
55 Pardee Place
New Haven, Conn. 06515
"Promote greater literacy on Apple II, Publish Newslad (ASAP) and aid new users."

# THREE GOOD REASONS YOU SHOULD READ

# COMPUTE. The Journal for Progressive Computing.™

## ISSUE 1, FALL, 1979


Selecting and Developing Small Business Systems—Potential Problems & Pitfalls ... *Mike Sawyer*, 4
Sorting Sorts: A Programming Notebook, ... *Rick & Belinda Hulon*, 7
Len Lindsay Reviews Three Word Processors—An Overview ... 13
Commodore Business Machines ... 14
Connecticut Microcomputer ... 17
Programma International ... 19
Microcomputers for Nuclear Instrumentation, ... *J.S. Byrd*, 24
Tokens Aren't Just For Subways: Microsoft Basic ... *Harvey Herman*, 29
Universal 6502 Memory Test PET, Apple, Sym and Others ... *Carl Moser*, 32
Microcomputers in Education ... *Pierre Barrette*, 33
Flying With PET Pilot: Kids and Microcomputers At Peninsula School, ... *Katie and David Thornburg*, 40
Teachers, Computers, and The Classroom, ... *C.J. Carr and Everett Carr*, 42
Atari Computers: The Ultimate Teaching Machines? ... *John Victor*, 62
The Evolution Of A Magazine ... *Len Lindsay*, 65
Pet In Transition — ROM Upgrade Map, ... *Jim Butterfield*, 68
A Commodore Perspective ... *Bob Crowell*, 71
Retrofitting ROMs ... *Larry Issacs*, 76
PET Screen Print Routine ... *David Malmberg*, 78
TRACE For The PET ... *Brett Butler*, 84
32K Programs Arrive ... *Len Lindsay*, 86
Using Direct Access Files With The Commodore 2040 Dual Drive Disk ... *Chuck Stuart*, 93
Mastering The Ohio Scientific Challenger 1P, ... *Keith Russell & Dave Schultz*, 97


## ISSUE 2, JANUARY/FEBRUARY, 1980


The Editor's Notes ... *Robert Lock*, I-3
The Consumer Computer ... *Len Lindsay*, 1
Interview with Dr. Chip ... *COMPUTE Staff*, 7
Sorting Sorts, Part 2 ... *Belinda and Rick Hulon*, 11
Memory Partition of Basic Workspace ... *Harvey B. Herman*, 18
Home Accounting, Plus An Easier Method of Saving Data ... *Robert W. Baker*, 23
Word Processing, A User Manual of Reviews - Part 2 ... *Len Lindsay*, 29
Book Review: 6502 Assembly Language Programming ... *Jim Butterfield*, 36
Machine Language Versus Basic: Prime Number Generation/AIM 65 ... *Marvin L. De Jong*, 39
Basic Memory Map (Page 0): Aim, Kim, Sym, PET, Apple ... *Jim Butterfield*, 41
Ramblin' ... *Roy O'Brien*, 42
The Learning Lab ... *Marlene Pratto*, 43
Micros and the Handicapped ... *The Delmarva Computer Club*, 44
**The Apple Gazette** ... 65
A Printer for the Apple: The Heath H14 ... *Don Earnhardt*, 66
**The Atari Gazette** ... 70
Comparison of Microsoft PET Basic with Atari Basic ... *Joretta Klepfer*, 70
The Ouch in Atari Basic ... *Glen Fisher and Ron Jeffries*, 75
Atari Basic, Part 2 ... *John Victor*, 76
**The Pet Gazette** ... 78
Computer Programs and Your Ethics ... *Robert Lock*, 78
The Programmer's Corner ... *Robert Lock and Larry Isaacs*, 80
Lower Case Descention on the Commodore 2022 Printer ... *W.M. Bunker*, 81
Saving Memory in Large Programs ... *Mike Richter*, 82
The Deadly Linefeed ... *Jim Butterfield*, 82
Yes, Nova Scotia, There is a Four ROM PET ... *Robert Lock*, 82
Apparent Malfunction of the < Key ... *Jim Butterfield*, 82
Using Direct Access Files with the Commodore 2040 Disk Drive ... *Chuck Stuart*, 87
Null Return Simulation for PET Users ... *Matsumoto, Weinshank and Davis*, 90
A Few Entry Points, Original/Upgrade ROM ... *Jim Butterfield*, 93
Plotting with the CBM 2022 Printer ... *Len Lindsay*, 93
Inside the 2040 Disk Drive ... *Jim Butterfield*, 94
PET Programs on Tape Exchange ... *Gene Beals*, 96
Review: Batter Up ... *Carl Strobel*, 98
Review: Plexivue ... *Art Hunkins*, 99
Review: Textcast ... *Joe Dilts*, 100
**The SBC Gazette** ... 102
The Single-Board 6502 ... *Eric Rehnke*, 102
A Sym-1 Message Scroller ... *A.M. MacKay*, 108
Adapting Basic Programs from other Machines to the OSI ... *Charles L. Stanford*, 110
Proofread for your KIM ... *Ralph Kelley*, 112
Notes on the Pulse-Counting Mode of the 6522 ... *Marvin L. De Jong*, 114
Tokens in OSI Basic ... *Barry L. Beal*, 116


## ISSUE 3, MARCH/APRIL, 1980


The Editor's Notes ... *Robert C. Lock*, 3
Computers and Society ... *David D. Thornburg and Betty J. Burr*, 7
Dr. Chip ... 8
Preview of Commodore's New Disk Basic 4.0 ... *Chuck Stuart*, 10
Enhancing Commodore's Word Pro II ... *Larry Isaacs*, 15
File Conversions on the Commodore 2040 Drive ... *Hal Wadleigh*, 18
Program listings for COMPUTE ... 20
Using the GET Statement on the PET ... *Alfred J. Bruey*, 23
Proper Diskette Handling ... *Chuck Stuart*, 24
Machine Language ... *Jim Butterfield*, 26
Review: Synertek Systems KTM-2 ... *Edward D. James*, 29
Utinsel: Enabling Utilities ... *Larry Isaacs*, 34
Identify Your Atari Colors ... *Len Lindsay*, 39
Manual Alphabet Tutorial on a PET ... *Susan Semancik*, 41
The Learning Lab ... *Marlene Pratto*, 51
Review: The Prestodigitizer ... *Laura M. Benson*, 56
A Micro for the Teacher ... *Thorwald Esbensen*, 58
Light Pen Selection on PET/CBM Screen ... *David R. Heise*, 60
The Consumer Computer ... *Len Lindsay*, 63
**The Apple Gazette** ... 68
Naming Apple Cassette Files ... *D.P. Kemp*, 68
On Apple II to Heathkit H-14 ... *Mike Wiplich*, 69
New Product Feature: 80 Columns and Upper/Lower Case Letters for Apple II ... *Michael S. Tomczyk*, 70
Apple Software Note ... *Eric Rehnke*, 72
**The Atari Gazette** ... 74
Notes on Atari ... *Robert Lock*, 74
Star Raiders Defend the Galaxy ... *Joretta Klepfer*, 74
Star Raiders: The Wizard Behind the Game ... *Michael S. Tomczyk*, 75
Review: Atari Basketball Cartridge ... *Len Lindsay*, 76
Atari Program Saving ... *Len Lindsay*, 76
Review of Atari's 810 Disk System ... *Ron Jeffries and Glen Fisher*, 78
**The PET Gazette** ... 81
Oops! ... *W.M. Bunker*, 81
Null Return Revisited ... *Earl H. Wuchter*, 81
Cheep Print, PART 1 ... *C.A. Mc Carthy*, 82
Direct Screen Input ... *Len Lindsay*, 88
No CB2 Sound? ... *Larry Isaacs*, 88
A Versatile Serial Printer Interface for the PET ... *Harvey B. Herman and Charles B. Pate*, 90
Rambling ... *Roy O'Brien*, 92
Review: MAE, A PET Disk-based Macro Assembler ... *James Strasma*, 93
Review: The PET Rabbit ... *James Strasma*, 94
PET Programs on Tape Exchange ... *Gene Beals*, 95
Memo to Machine Language Programmers ... *Jim Butterfield*, 96
**The SBC Gazette** ... 98
The Single Board 6502 ... *Eric Rehnke*, 99
More about compute II ... *Robert Lock*, 104
Nuts and Volts ... *Gene Zumchak*, 105
Read PET Tapes with your AIM ... *Mark Reardon and Eric Rehnke*, 110
Review: KIMEX-1 ... *Harvey Herman*, 113
Fast Tape Read/Write Programs for your OSI ... *Edward H. Carlson*, 115
Applications Review: Logic Analyzer for KIM ... *Roy Flacco*, 118
Try Pseudo Code ... *Eric Rehnke*, 120


# ONE GOOD REASON TO READ

# compute II.

## The Single-Board COMPUTE.™

## ISSUE 1, APRIL/MAY, 1980


The Editor's Notes ... *Robert Lock*, 2
The Single-Board 6502 ... *Eric Rehnke*, 3
Nuts & Volts ... *Gene Zumchak*, 9
RS-232 Communications, Part 1 ... *Michael E. Day*, 16
An Upgrade for KIM Microchess 1.0 ... *Garold R. Stone*, 19
Program Transfers (Pet to Kim) ... *J.A. Dilts & H.B. Herman*, 25
Designing an IEEE-488 Receiver With The SYM ... *Larry Isaacs*, 27
Fun With the 1802 ... *Larry Sandlin*, 34
Improved Pulse Counting Software for the 6522 VIA ... *Marvin L. DeJong*, 36
Printing A Symbol Table for the AIM-65 Assembler ... *Richard F. Olivo*, 40
Hard Copy Graphics for the Kim ... *Keith Sproul*, 43
24 Hour Clock for SYM-1 Basic ... *A.M. Mackay*, 46
Screen Clear Routines for the OSI C1P ... *Charles L. Stanford*, 49
Machine Language Tapes for OSI Challengers ... *Daniel Schwartz*, 52
Songs in the Key of KIM ... *George W. Hawkins*, 54


# The 6502 Resources

# COMPUTE. and compute II.

6 Issue (annual) subscription: U.S. $9.00, CANADA $12.00 U.S. Important Note: Beginning with Issue 4, **COMPUTE** covers Pet, Apple and Atari. The Single Board Computer Gazette moves to our new magazine, **compute II. The Single-Board COMPUTE,** covering Sym, KIM, AIM, OSI, and 1802 based machines. 6 Issue (annual) subscription: U.S. $9.00, CANADA $12.00 U.S. **Subscribe to both through June 30, 1980** for $15.00, U.S.; $18.00 U.S. in CANADA. **COMPUTE. P.O. Box 5119, Greensboro, N.C. 27403, (919) 272-4867** Publications of Small System Services, Inc., 900-902 Spring Garden Street, Greensboro, N.C. 27403.

# The MICRO Software Catalog: XX

**Mike Rowe**
**Box 6502**
**Chelmsford, MA 01824**

Name: **The Designer System**
System: **Apple II or Apple II +**
Memory: **48K**
Language: **ROM APPLESOFT**
Hardware: **Apple II w/DISK II**

Description: The Designer is a HIRES graphics macro-operating system that provides the user with line and curve creation with game paddles (or Joysticks) and single keystroke ease. Lines, circles, arcs, ellipses, rectangles, areas, etc. may be quickly drawn, modified, and saved to Disk as completed or unfinished drawings. Both HIRES pages are used to provide 2 position animations. Typical uses are computer art, graphic game setups, visual presentations, and showing off. Sometimes called "the poor man's graphics tablet" this program does your complex hplotting for you. ERROR-FREE-GUARANTEE

Price: **$24.95**
Includes: Disk with DEMOS and Manual, guarantee
Author: **Jeff Johnson, Apple Jack**
Available: Your dealer or Apple Jack
12 Monterey Drive
Cherry Valley, MA 01611

Name: **ACTS**
System: **Apple II**
Memory: **32K RAM with ROM Spplesoft**
Language: **Applesoft and Machine Language**
Hardware: **Apple II, Disk II, D C Hayes Micromodem.**

Description: The Apple Communication Transfer System (ACTS) and an Apple (equipped with a disk drive, ROM Applesoft, and a C C Hayes micromodem) will transfer over the telephone Apple programs in all three languages. Exchange programs with others without leaving your home. No program modifications, self adapting and easy to use. Automaticlly stores the transfered program on the receiving Apple's disk, ready for use. The entire ACTS system, on a disketter with complete documentation, retails for only $14.50. All proceeds derived from the sale of ACTS will go toward the procurment of micro hardware for the Northeast Ohio Apple Bulletin Board System.

Copies: **30 plus**
Price: **$14.50 on disk**
Includes: System diskette and full documentation
Author: **Northeast Ohio Bulletine Board System**
Available: The NEO/ABBS
P.O. Box 4731
Clevelnd, Ohio 44126

Name: **Road Race**
System: **Apple II**
Memory: **16K min**
Language: **Integer Basic and Machine Language**
Hardware: **Game paddles or joysticks**

Description: Real-time simulation of Grand Prix Road Racing. Two players race around a 2.25 mile course, or one player races against a computer driven car. HIRES display shows through-windshield view of race course.

Price: **Cassette $15.00, disk $20**
Author: **Stan Erwin**
Available: **Stan Erwin**
**5410 W. 20th Street**
**Indianapolis, IN**
**46224**

**Name: Space Shuttle Landing Simulator**
System: **Apple II**
Memory: **48K**
Language: **Machine language and Applesoft**

Description: Slightly improved version of program advertised in November 1979 of MICRO. Give system config.

Copies: **250 plus**
Price: **Applesoft RAM $15.00 on cassette, Applesoft ROM $17.00 on casette, Diskette version $21.00. State which.**
Author: **John Martellaro**
Available: Harvey's Space Ship Repair
P.O.Box 3478 Univ. Park
Las Cruces, NM 88003

Name: **Restaurant Evaluation**
System: **Apple II**
Memory: **16K**
Language: **Applesoft II**
Hardware: **Disk II, Printer (both optional)**

Description: Evaluates potential restaurant/night club sites and thereby reduces the margin of risk involved in purchasing a new or existing business. All the necessary percentages and formulas are programmmed to evaluate whether a potential site will be profitable or not. The program is also structured for use by present restaurateurs to evaluate whether or not their present business is operating at cost and profit effeciency. Calculates montly gross, computes monthly loan rates (or mortgage), and reports weekly, monthly and annual net profit/loss in dollar amounts and percentages.

Copies: **25 +**
Price: **$19.95 Diskette plus $1.95 P&H First Class Mail, Check or Money Order.**
Includes: Diskette and full documentation
Author: **M. Goldstein**
Available: Mind Machine, Inc.
31 Woodhollow Lane
Huntington, N.Y. 11743

Name: **Trace/Debug-Monext**
System: **SYM-1**
Memory: **2K (for cassette version)**
Language: **Assembler**
Hardware: **Standard SYM (w/CRT)**

Description: This program adds 15 commands to SYM's monitor including: Trace, Disassemble, Relocate, Find, ASCII dump, Stack dump, etc. The "T" command sets up its own operating environment supporting commands such as, Go, Skip, Continue, Single Step, Memory/Register examine/modify, ect. As SYM executes each instruction of the user program, an NMI is generated. IF the address of the instruction is "valid" — neither in SYM's monitor not the extension — and if it is not is a "skip" range, a disassembly/register listing is printed. This program as a whole is clean and operates transparently under SYM's OS. SASE for complete specs and examples.

Copies: **Just released**
Price: **Object listing**
**$14.95**
**Cassette**
**$15.95 @$3800 or specify EPROM (2716)**
**$49.95 @$F000 or specify Commented Source**
**$9.95**
Author: **Jeff Holtzman**
Available: Jeff Holtzman
6820 Delmar-203
St. Louis, MO 63130

Name: **LEM LANDER**
System: **Apple II**
Memory: **32K**
Language: **Applesoft**
Hardware: **Disk II**

Description: Lem Lander is a real-time version of the popular lunar lander game. This disk-based game includes nine landscapes to try your hand at landing on. Your high-resolution LEM is controlled through space via the paddle knowb (thrust) and the buttons (rotation).

Copies: **One for you**
Price: **$14.95**
Author: **Barry Cox**
Available: Barry Cox
444 Myers Avenue
Harrisonburg, VA 22801

Name: **UTIL-DS**
System: **Apple II**
Language: **Machine language and Applesoft**
Hardware: **Apple II**

Description: UTIL-DS is a collection of several machine language utility routines and one Applesoft utility routine. The Applesoft utility is a sophisticated formatting routine for numeric output. The routine converts numeric values into a character string for printing. The user of the routine specifies the maximum length of the resulting string and the number of decimal places to appear in the result. Positive and negative numbers can be converted by the routine. Comma are inserted in the integer portion of the number. The machine language utilities consist of several routines to improve the error handling capabilities of Applesoft programs (e.g. resume execution at the statement following the one in error), a machine language to Applesoft interface utility, a routine to selectively clear arrays and a routine for loading machine language programs into RAM along with an Applesoft program.

Copies: **Just released**
Price: **$35.00 (Texas residents add 5% sales tax)**
Includes: Routines on diskette, a sample program to demonstrate numeric formatting and documentation.
Author: **Robert F. Zant**
Available: Decision Systems
P.O.Box 13006
Denton, TX 76203

Name: **Dynatext Editor**
System: **PET/CBM, ROM**
Memory: **16K or more**
Language: **Basic, plus machine-language repeat key**
Hardware: **Commodore 2022 or 2023 Printer (optional)**

Description: Authorized PET version of "Context Editor", as printed in Kilobaud Magazine 5/79. Enhanced and changed in many ways for the PET. Uses cassette or disk. Has all the desirable features of most good word processors, such as global search and replace, right justification, cursor editing, etc. Plus dynamic formatiing, the ability to print in any desired shape. Holds 7 pages of text at once in a 32K PET.

Copies: **5, Just Authorized**
Price: **$5.00 for cassette, program and instructions.**
Author: **James Strasma, based on work by Law & Mitchell**
Available: Rev. James Strasma
120 West King Street
Decatur, IL 62521

Name: **Higher Graphics II**
System: **Apple**
Memory: **32K and disk drive**
Hadware: **Apple I**

Description: A collection of programs and shape tables that lets any programmer create detailed and beautiful high resolution displays and animation effects. Make your programs come alive by utilizing the full graphical capabilities of the Apple II. The package contains:

Shape Maker - create shapes with this easy to use shape table generator. Start new shape tables or add to existing ones. Correct shapes as they are being produced. Delete unwanted shapes from the table. Display any/all shapes with any scale or rotation at any time.

Table Combiner - pull shapes from existing general purpose tables and add the ones you want into a new special purpose table. May combine shapes from any number of tables. All shapes can be viewed or deleted.

Screen Creator - place your shapes on the hig-res screen. Add areas of color and text to make detailed displays or game boards for high resolution games. A screen can be created in minutes with this easy to use program. Utilizes any number of shape tables and allows screen to be saved at any time.

Shapes - four shape tables with over 100 shapes are provided. Included are alphanumerics, chess figures, card symbols (club, spade, etc), tanks, planes, spaceships, ships, cars, trees, mountains, buildings, etc. Add the shapes you like to your own table.

High Res Text - how to use high resolution graphics in your program. Animation effects and display techniques.

Price: **$24.00 Retail**
Available: Synergistic Software
5221 120th Avenue SE
Bellevue,WA 98006

Name: **HYPNOSIS**
System: **Apple-1 disk drive**
Memory: **32K**
Language: **Integer Basic**

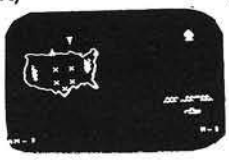
Description: Hypnosis is a program that uses Apple's video and sound capabilities to aid in suggestive relaxation, behavior modification and trance induction. Visual and auditory patterns are fully variable for shape, color and frequency matching of the subject's alpha brain wave rhythm. Designed for health professionals and students of the medical, psychological and social sciences.

Copies: **250 plus**
Price: **$20.00**
Includes: Diskette, program and manual
Author: **E.J. Neiburger**
Available: Andent Inc.
1000 North Avenue
Waukegan, IL 60085

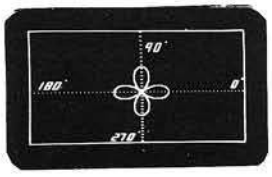
# PROGRESSIVE SOFTWARE

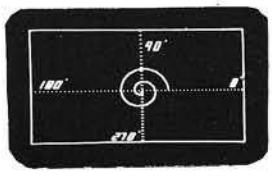
## Presents
## Software and Hardware for your APPLE

**Missile—Anti—Missile (Aplsft)** **$9.95**



**Polar Coordinator Plot** **$9.95**






by TD Moteles

**Tables Generator** forms shape tables with ease from directional vectors-starting address, length and position of each shape. Program saves shape tables anywhere in usable memory.
Murray Summers **$9.95**

**Sales Forecast** provides the best forecast using the four most popular forecasting techniques: linear regression, log trend, power curve trend, and exponential smoothing.Neil D. Lipson **$9.95**

**Single Drive Copy** is a special utility program, written by Vince Corsetti in Integer BASIC, that will copy a diskette using only one drive. It is supplied on tape and should be loaded on to a diskette. It automatically adjusts for APPLE memory size and should be used with DOS 3.2 **$19.95**

**Curve Fit** accepts any number of data points, distributed in any fashion, and fits a curve to the set fo points using log curve fit, exponential curve fit, least squares, or a power curve fit. It will compute the best fit, or employ a specific type of fit, and display a graph of the result. By Dave Garson.**$9.95**

**Touch Typing Tutor** teaches typing. Indicates speed and errors made. Finger Bldrs, Gen. Typing, Basic Language and User Supplied. Diskette. Written by Wm. A. Massena. **$19.95**

**Apple Menu Cookbook** index-accessed data storage/retrieval program. Recipes stored, umlimited lines per entry. Easy editing. Formulated after N.Y. Times Cookbook. Other useful features included. Written by Wm. Merlino, M.D. **$19.95**

**Mailing List Program** maintains complete record of name, address, phone no., mailing lables accommodated parallel card or built-in printer driver,easy data entry
Diskette. 32K. **$19.95**

**POSTAGE AND HANDLING**

Please add $1.25 for the first item and $.75 for each additional item.

- **Programs accepted for publication**
- **Highest royalty paid**

**Utility Pack** combines four versatile programs by Vince Corsetti, for any memory configuration.

- **Integer to Applesoft conversion:** Encounter only those syntax errors unique to Applesoft after using this program to convert any Integer BASIC source.
- **Disk Append:** Merge any two Integer BASIC sources into a single program on disk.
- **Interger BASIC copy:** Replicate an Integer BASIC program from one disk to another,as often as required, with a single keystroke.
- **Applesoft Update:** Modify Applesoft on the disk to eliminate the heading always produced when it is first run.
- **Binary Copy:** Automatically determines the length and starting address of a program while copying its binary file from one disk to another in response to a single keystroke. **$9.95**

**Solitare —** Old European peg game, played by one (similar to Chinese checkers). Object — to finish with last peg in center.
Charles B. Smith **$9.95**

**Water The Flowers —** Math (add., subt., mult., div,.) grades 1-6. (disk)
Judy Pegg **$19.95**

**Catch The Pig —** teaches the cartesian coordinate system.Educ. game, upper grade school level, 1-4 students, many levels of play. Disk or tape.
Judy Pegg **$16.95** (disk) **$9.95** (tape)

**Blockade** lets two players compete by building walls to obstruct each other. An exciting game written in Integer BASIC by Vince Corsetti. **$9.95**

**Saucer Invasion, Space Maze, Starwars, Rocket Pilot:**
Written by Bob Bishop **Each $9.95**

### Hardware

**Light Pen** with seven supporting routines. The light meter takes intensity readings every fraction of a second from 0 to 588. The light graph generates a display of light intensity on the screen. The light pen connects points that have been drawn on the screen, in low or high resolution, and displays their coordinates. A special utility displays any number of points on the screen, for use in menu selection or games, and selects a point when the light pen touches it. The package includes a light pen calculator and light pen TIC TAC TOE. Neil D. Lipson's programs use artificial intelligence and are not confused by outside light. The HIRES light pen, only requires 48K and ROM card **$34.95**

## TO ORDER

Send Check or Money Order to:

**P.O Box 273**
**Plymouth Meeting, PA 19462**

PA residents add 6% sales tax.

**U.S. and foreign dealer and distributor inquires invited**
**All programs require 16K memory unless specified**

# 6502 Bibliography: Part XX

**Dr. William R. Dial**
**438 Roslyn Avenue**
**Akron, OH 44320**

**594. MICRO 17 (October 1979)**

Peck, Robert A., "SYM-1 6532 Programmable Timer," pgs. 55-56.
The 6532 programmmable timer is useful as a backup timer or as a loop controller.

Kintz, Robert T., "A Real-Time Clock for OSI Disk Systems," pgs. 59-60.

Dial, William R., "6502 Bibliography: Part XIII," pgs. 61-62.
About 70 more references on the 6502.

**595. Applesauce 1, No. 1 (March 1979)**

Ohrbach, Jeffrey,"The 6502 is Stacked," pgs. 5-6.
Discussion toward understanding those OP-Codes affecting the machine stack of the Apple.

Wigginton, Randy, "Apple's Bootleg Assembler," pgs. 11-14.
Documentation for the program 'Randy's Text Editor and Weekend Assembler (Ted II)."

**596. Applesauce 1, No. 2 (April 1979)**

Hyde, Randall, "Loaded and Gone," pgs. 5-6.
How to make your own "load and go" tapes for the Apple.

Emrich, Dick, "Simple Sort," pg. 8.
A simple bubble sort for the Apple.

Curtis, Frank, "Numerical Control Programming," pgs. 11-12.
Related to the control of machine tools.

Baker, Dwight, "The Assembly Line," pgs. 15-16.
A tutorial using Randy's Assembler.

Anon., "Docu-Prog: Lazer's Text Editing System," pgs. 16-18.
Documentation for the Lazer Text Editor.

**597. Applesauce 1, No. 3 (May, 1979)**

Walls, William, "Yes, Virginia, You CAN Save Data," pgs. 4-6.
A tutorial on Data Storage Methods.

Hyde, Randy, "The Apple Monitor—Part 1," pgs. 8-9.
Dealing with the many uses of the Apple Monitor.

Irvine, Al, "Docu-Prog: Program Compare," pgs. 10-11.
Documentation for PROGRAM COMPARE which compares the listings of two programs.

Paymar, Dan, "A Disc Write Protect/Enable Override Switch," pg. 12.
A simple hardware modification for your Apple Disc.

Anon., "Charts," pgs. 14-16.
Apple II ASCII CHART; Apple Integer Basic HEX Representation-Numerical Order; Hex Representation Chart-Alphabetical; Vector Table Address Chart.

**598. Applesauce 1, No. 4 (June 1979)**

Curtis, Frank, "Dr. Memory—A New Text Processor," pgs. 5-6.
A review.

Hyde, Randall, "Single Disk Text File Transfer," pg. 8.
Transfer a text file with only one disk drive.

Anon., "LISA: What is an Interactive Assembler Anyway?" pgs. 8-9.
A review of LISA.

Lu, Ron, "Docu-Prog: Beneath Apple Manor," pgs. 10-11.
Documentation and Review of "Beneath Apple Manor," an adventure type game.

Anon., "Apple II Mini-assembler F666G," pg. 12.
How to use the mini-assembler.

Haefner, Mike S., "Processor Status Register 'P'," pgs. 13-16.
Discussion of the processor status register and chart of flags.

Spurlock, Loy, "Hex Representation Chart," pg. 13.
Discussion of the Apple Token Chart.

Anon., "Zero Page Chart," pg. 14.
Discussion of the contents of the Apple page zero.

Anon., "Apple II 6502 OP CODE Chart," pg. 15.
Chart showing the codes for different types of addressing on the 6502.

Anon., "Integer Basic Internals," pg. 20.
List of variables used by basic; integer basic routines.

Anon., "Useful Basic Pointers," pg. 22.
Discussion and list of addresses.

Tognazzini, Bruce, "Apple Basic Interpreter Instruction Set," pg. 22.
Names and addresses.

Anon., "Apple DOS Symbol Table," pg. 27.
Names and addresses.

**599. Applesauce 1, No. 5 (July/August 1979)**

Lemens, Vernon Jr., "Superchip Eccentricities," pgs. 8-13.
A tutorial on the Superchip for the Apple.

Anon., "New Product Releases," pgs. 10-11.
Description of the Dan Paymar lowe case chip, the Universal Parallel card, the Apple bulletin board system, etc.

Anon., "Original Apple Corps' Program Library," pgs. 16-17.
Program library.

Lenz, John and Spurlock, Loy, "From the Forum," pgs. 20-22.
A few Calls and JSR's for the Apple.

Diay, Robert, "The Assembly Line: Page List Program," pgs. 23-24.
Page List is a machine language routine to list twenty lines at a time.

Amromin, Joel L., "Apple II Slowlist," pgs. 24-25.
Paddle-controlled slowlist program.

Hyde, Randy, "UCSD PASCAL," pg. 29.
A short tutorial on PASCAL for the Apple.

**600. Byte 4, No. 11 (November 1979)**

Partyka, Dave, "Shape Table Conversion for the Apple II," pg. 63.
Hints for using shape tables.

Govind, P.K., "Interfacing the PET to a Line Printer," pgs. 98-102.
Interfacing the 8-bit user port on the PET 2001, including software program "PRINTSCREEN."

**601. Creative Computing 5, No. 11 (November 1979)**

Heuer, Randy, "PET Software from Creative Software," pg. 46.
Five programs are reviewed.

Daro, Paul, "A Home Control System," pgs. 54-59.
Description of the use of the Introl X/10 system marketed by Mountain Hardware.

Yob, Gregory, "Personal Electronic Transactions," pgs. 183-185.
This month's column discusses Software in ROM; some graphics gizmos; animation routines, etc.

**602. Stems from Apple 2, Iss. 10 (October 1979)**

Hoggatt, Ken, "Ken's Korner," pg. 3.
A short tutorial on Pascal with sample routines.

Keyes, Pat, "ROM Applesoft II Vector Chart," pgs. 4-5.
Machine language subroutines and keywords.

Newman, Will II, "File Handler," pgs. 6-9.
Routine for handling files.

**603. MICRO, No. 18 (November 1979)**

Wells, George, "Dual Tape Drive for SYM-1 Basic," pgs. 5-7.
Make your SYM-1 Basic work with two tape recorders and manage tape cassette files.

Murphy, S. R., "Some Useful Memory Locations and Subroutines for OSI Basic in ROM," pgs. 9-10.
Where some important subroutines reside in OSI Basic.

Hawthorne, Alan R., "A Tape Indexing System for the PET," pgs. 11-13.
Rapid indexing of the PET cassette.

Swanson, Mark, "Subroutine Parameter Passing," pgs. 14-15.
A technique to facilitate passing parameters to subroutines.

Bishop, Bob, "Apple II Hires Picture Compression," pgs. 17-24.
How to put several times as many hires slides on a single disk as previously done. Used in Bob Bishop's Super Slide Show.

Floeter, Alan D., "Assembly Language Applesoft Renumber," pgs. 27-29.
A very fast renumber program.

Bruey, Alfred J., "Performing Math Functions in Machine Language," pgs. 30-31.
Math for the KIM-1.

Hooper, Philip K., "TSAR: A Time Sharing Administrative Routine for the KIM-1," pgs. 35-41.
Tsar is a super monitor which supports time-sharing for the KIM.

Dennis, Jim, "Interfacing the CI-812 to the KIM," pgs. 43-44.
The Percom CI-812 contains a full-duplex data terminal interface and a fast cassette (2400 baud).

Hill, Alan G., "Ampersort," pg. 45.
A corrected listing for this fast sort routine.

Leary, Richard A., "SYM-1 Baudot TTY Interface," pgs. 49-54.
Teletype with your SYM-1.

Rowe, Mike (staff), "The MICRO Software Catalog: XIV," pgs. 55-56.
Six new programs for 6502 devices are reviewed.

Irwin, Paul, "Alarming Apple," pgs. 59-60.
Teach your Apple to respond to errors with an alarm and keyboard lockout.

Dial, William R., "6502 Bibliography: Part XIV," pgs. 61-62.
Fifty-five more references in the 6502 literature.

**604. Control Engineering (October 1979)**

Faust, Gregory, "Programmable Controller Offers Fiber Optic Data Link for Remote I/O," pgs. 53-54.
Both the fiber optic interface and the remote I/O use a 6502 microprocessor for communication purposes.

**605. Rainbow 1, Iss. 9 (October 1979)**

Fleming, Jim, "Updating the ARESCO DMS," pgs. 7-16.
An improved file management system.

Vermehr, Jay, "An Apple Recorder Box," pgs. 19-20.
A very useful box to use between the Apple and the tape-deck.

Anstis, Stuart M., "Rocket Lander in Lo-Res Graphics," pgs. 21-24.
A game in Apple II Applesoft.

**606. The Paper 2, Iss. 8 (October 1979)**

Wachtel, A., and Szepesi, Z., "Pythagorean Triplets Revisited," pgs. 12-17.
A program for the PET.

Costarakis, Dennis A., "A Screen Dump Subroutine for Use with the 2022/2023 Printers," pgs. 23-24.
Routine for the PET.

**607.Personal Computing 3, No 12 (December 1979)**

Forbes, John L., "Applesoft Conversions," pg. 10
Conversion of a program in PT Basic to Applesoft Basic.

Whack, Margaret, "Create Your Own Periodical Guide," pgs 69-70.
Catalog your favorite computer magazine articles.

**608. Recreational Computing 8, No 3 (Nov/Dec 1979)**

Carpenter, Chuck, "Apple II's Three M's," pgs. 28-31.
More on Memory, Monitor and Machine language.

Bruey, Alfred J.,"Making Music on the PET," pgs. 49-51.
Software and hardware for a musical PET.

**609. Kilobaud Microcomputing No 36 (December 1979)**

Lindsay, Len, "Pet-Pourri," pg. 12.
Discusses Commodore Word Processor and Printer, Games, Graphics programs.

Lancaster, Don, "Lowercase for Your Apple II," pgs. 34-42.
Now for the software to complete this two-part article.

Haehn, Lou, "Chess I for the Apple II," pgs. 46-52.
Writing a chess program for the Apple II.

Lary, Richard A.,"Reverse Video from OSI's 540 Board," pgs. 128-129.
Black on white to enchance graphics displays.

David, D. J.,"PET's Machine Language Monitor," pgs. 134-140.
A review and analysis of the PET Monitor.

**610. Call Apple 2, No 8 (October 1979)**

Cahill, Gerald, "Auto Number," pgs. 5-8.
A program to automatically number Applesoft programs as they are input.

Foote, Gary A., "Multiple Disk Catalog," pg. 11.
A special program to read catalogs off Apple disks for entry into File Cabinet.

Merchant, Fred, "Instructions for Running Multiple Disk Catalog," pg. 13.
Detailed instructions for running Diskcat.

Barnes, Keith Allen, "File Cabinet Improvements," pgs. 13-14.
Fixes to make this program work better on the Apple.

Golding, Val J., "So Who Needs Applesoft (Revisited)," pgs. 15-16.
A tutorial for the Apple including INT BASIC VAL FN SUB-ROUTINE.

Corsetti, Vince, "Self Writer," pg. 18.
This program performs the equivalent of an exec file without using DOS.

Reynolds, Lee, "Logical Variables in Apple II Basic," pgs. 21-22.
A tutorial.

Golding, Val J., "Exec File Shortcut," pg. 24.
A short program to create a text file that when EXEC'd would BLOAD a binary program, set HIMEM, and Run a basic program.

Golding, Val J., "Flash Card Modifications," pgs. 24-25.
How to provide different data sets from a disk to run the Flash Card Program.

Krantz, Bill, "Write -Apple," pgs. 26-27.
All about using the High Speed Serial Card with Integral Data IP 125/225 printer.

Hockenhull, James L., "Simulating a Two-Dimensional Matrix from Integer Basic," pgs. 30-31.
A new way of handling Hi-Res programs.

Alex, Steve, "Applesoft II Firmware Card Hi-Res Routines," pgs. 33-34.
How to use the hi-res graphics routines available on the Applesoft II firmware card from assembly programs.

Garson, David B., "Poor Man's PRINT—USING Program," pg. 34.
A useful utility routine.

**611. Fort Worth Apple Users Group 1, No. 4, (October 15, 1979).**

Meador, Lee, "Interrupt Daisy Chain," pg. 2.
An explanation of daisy chain interrupts.

Meador, Lee, "Disassembly of Dos 3.2," pg. 4-14.
Disassembly program.

**612. Stems from Apple 2, Issue 11 (Nov. 1979)**

Dolema, Nels, "Worm," pg. 4.
A new version of the "WORM" with paddle control.

**613. Abacus II Newsletter I, Iss. 11 (Nov. 1979)**

Anon, "DOS 3.2 R.H.D.," pg. 2.
Addresses and functions of DOS 3.2 for Apple.

Howard, Clifton M., "Hex/Dec Conversion Chart," pg. 5.
Poor Man's Ti programmer.

Howard, Clifton M., "Diablo Character Values,"pg. 6.
Useful table for Apple owners using the Diable printer.

Howard, Clifton M., "Interpretation of Memory," pg. 7.
Apple II Basic Integer Memory Interpretation.

Freeland, Bruce, "Plotting Algorithm," pg. 8.
Apple II page I layout and map.

Yee, David R., "Stop that Blinking Cursor,;; pg. 12.
Now you can jazz up your own programs with a custom Cursor.

Anon, "Disk Access Update," pg. 12.
A fix for a previous routine.

Staff, "Control Character Show," pg. 12.
Show up those hidden control characters.

**614. The Target, (Nov/Dec 1979)**

Anon, "Basic Short Cut," pg. 2-4.
This routine automatically inputs characters to the Basic interpreter thus freeing the user of some drudgery of entering programs, into the AIM 65.

Butterfield, Jim, "Inside Basic-Tokens," pg. 4-8.
Key addresses, zero page usage, ROM organization, etc. for the AIM 65.

Bressen, Steve, "Roll," pg. 9.
Program to scroll a message onto the AIM-65 display.

**615. Southeastern Software Newsletter, No. 14, (Nov. 1979)**

Anon, "Length and Starting Address of Binary Files," pg. 4-5.
How to find binary program addresses in your Apple.

Anon, "El Cheapo Pascal Lower/UPPER CaseWriter," pg. 5-7.
A tutorial Pascal program.

Carpenter, Chuck, "Game Paddle I/O Applications," pg. 7-9.
An informative "How to" article.

**The Apple Shoppe 1, No. 4, (Nov. 1979).**

Van Winkle, Don, "Hi-Res Artillery Game," pg. 4-8.
A HiRes game of artillery, shooting over random terrains at the enemy gun.

Welman, Chuck, "Easywriter—Text Editor Review," pg. 10-11.
A rather favorable review of this text editor.

Anon, "Plotting Functions," pg. 12-14.
A tutorial for the Apple.

Welman, Chuck, "A Review of 'APPLE DOC'," pg. 15.
Useful Utility to aid programmers.

Anon, "String Magic," pg. 16.
All about strings.

Smith, David E., "Language Lab; Apple Pascal," pg. 17-22.
A tutorial on Pascal with program examples.

Crouch, Bill, "Apple Text Editors,;; pg. 23.
Reviews Applecations Unlimited Version 2.2, with Apple Pie Text Editor and The Word Weaver.

Crouch, Bill, "Apple II Text Processing System," pg. 27-28.
A review of this text editor and assembler.

**617. Appleseed Newsletter 1 No. 12 (Dec. 15, 1979)**

Hyde, Bill, "International Apple Core," pg. 1.
The International Apple Core is being formed as an organization to provide software, information and other services to the Apple user. Officers include Val Golding, Ken Silverman, Dave Gordon, Neil Lipson, etc.

**618. Apple Bits (Dec. 1979)**

Geier, Bob, "Basic Errors?," pg. 3.
How to rewrite your disk commands on the Apple.

Anon, "Rumor Mill," pg. 3.
The new Radio Shack TRS-90 will probabally be based on the new Motorola 6809 microprocessor. The Apple III may choose this chip or a new chip by MOS Technology. There is an unconfirmed rumor that Heath will discontinue production of the h-8.

**619. Byte 4, No. 11 (Nov. 1979)**

Anon, "Free Newsletter," (Nov. 1979)
Hands On! is a free newsletter published three times a year for science and technology educators and the initial issue contains an article "A Biased Introduction to the World of the 6502 Microprocessor."

**620. Fort Worth Apple User Group Newsletter, No. 5 (Nov. 15, 1979)**

Cahill, Gerald, "Auto Number," pg. 1.

Meador, Lee, "DOS Disassembly," pg. 2-16.
Listing of the assembly language for the DOS 3.2. Also a detailed listing and explanation of the RWTS Routine.

Meador, Lee, "Drawer-for Hi-Res Pictures," pg. 17-19.

Hoyt, Jim, "Special Subroutine," pg. 19.
Subroutine to allow prohibited characters like commas, etc. in string inputs.

**621. Contact No. 6 (Oct. 1979)**

Anon, "Invisible Writing," pg. 5.
How to plot one page of graphics while displaying the other.

Anon, "DOS Update for Dual Drive Users," pg. 6.
Improve the DOS 3.2 by updating to 3.2.1 on the Apple.

Anon, "Dollars and Cents," pg. 8-9.
Formats numeric output on the Apple to a dollar and cents format.

Anon, "Restore to Line Number," pg. 9-10.
A demo of how to do a RESTORE statement to a particular line number, on the Apple II.

Anon, "What Interface Card is in this System, Anyhow?", pg. 10.
With this program, CONFIG, it is possible to tell just what interface is in a particular slot.

**622. Rainbow 1, Iss. 10 (Nov. 1979)**

Deardon, Dr. Hinkley W., "From the Pits," pg. 15-16.
A two bit serial interface for the Selectric typewriter.

Wachtel, A. and Szepesi, Z., "The Development of a Basic-Program," pg. 9-11.
Illustrating the many ways in which a seemingly simple programming task on the Apple can be improved.

Laudereau, Terry L., "Pokeing Machine Language from Basic," pg. 17-18.
A tutorial article on entering machine language into the Apple II.

Wagner, Roger, "Fast Moves in Applesoft," pg. 19-20.
How to use the MOVE routine present in the Monitor by calling from Applesoft using a special routine.

Wagner, Roger, "One Less Error," pg. 19-20.
How to keep Apple's Integer Basic programs from bombing when dealing with an address greater than 32767.

**623. Apple Peelings 1, No. 3, (Nov. 1979)**

Anon, "November DOM (Disk of the Month)," pg. 3.
A dozen good programs plus Library List 10-79, the latest listing of the SF Apple Core's complete library as of the last of October, 1979.

Johnson, Allen, "What Is It?," pg. 4.
A short routine to identify the DOS 3.1 or 3.2 on Apple diskettes.

Fisher, Frank E., "Slot ÷s as Variables," pg. 4.
Specifying I/O slots in the Apple as variables and using in programs.

**624. The Paper 1, No. 3 (Nov. 1979)**

Lee, Arnie, "Clocks and Timers," pg. 4.
How to use the microprocessor clock in the PET to use in a timer or real time clock.

Anon, "Screen Display and Cursor Positioning," pg. 7-9.
A short tutorial for the PET.

Anon, "To Write a Character String to the Screen," pg. 9-10.
Short tutorials on this and a number of other short PET routines.

Anon, "PET USER GROUPS," pg. 15.
A list of about 40 user groups for the PET.

Szepesi, Zoltan, "Using the Monitor Subroutines," pg. 16-21.
A tutorial to show how the PET actually runs programs.

**625. Apple-Com-Post No. 4 (Nov. 1979)**

Anon, "Software Tups and Tricks," (in German), pg. 7.
All about handling numbers on the Apple.

**626. Call-Apple 2, No. 9 (Nov./Dec., 1979)**

Greenfarb, Sandy, "Internal Structure of Integer B asic," pg. 5-10.
A tutorial on Integer Basic.

Golding, Val, "Why Variables," pgs. 11-12.
Why replace numeric constants with variables in both Apple Integer and Applesoft.

Hyde, Randall, "The Assembly Line," pgs. 14-17.
Benchmarking Sweet 16 with 6502 Assembly Language. 6502 Machine code runs 5-7 times as fast as Sweet 16 but Sweet 16 code requires only about half as much memory to perform an equivalent function, on the Apple.

Sedgewick, Dick, "Sedgewick Plays it Straight," pgs. 19-20.
A tutorial dealing with the consequences of tokenizing, etc.

Golding, Val, "Basic Memory Move," pg. 23.
A hex memory move program for the Apple, and a corresponding one in Decimal.

Cox, Ross E., "Life with an Apple," pgs. 30-34.
All about how to make Life more enjoyable by improving The Game of Life on the Apple.

Hilger, Jim, "Apple Gaming: Playing Card Generation," pgs. 39-45.
General purpose routines that will generate hi-res images of playing cards.

**627. Stems from Apple 2, Iss. 12 (December 1979)**

Reinhardt, John, "President's Message," pg. 1.
An announcement about the newly formed "International Apple Core Club" to which individual User Groups may subscribe.

Stein, Greg, "Circles," pg. 3.
A short circle drawing program.

Anon., "The Twelve days of Christmas," pg. 3.
A program which prints the traditional Crhristmas song.

Ward, Dennis, "Do Your Words Runneth Over?," pg. 4.
How to avoid split words, bad spacing, etc.

Doeleman, Nels, "The Appl-Ogical Way to Arrange Numbers in DEcending Order," pg. 5.
Program to arrange numbers.

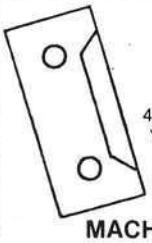
# SYM-1

INTERACTIVE TRACE/DEBUG
MONITOR ENTENSION COMMANDS

MONEX:***NEW COMMANDS:*Disassemble *Relocate *Find *ASCII dump *Trace (Sym-Bug) *Checksum calculator *BRK set/delete *More
SYM-BUG: Trace with dissassemble and register list; skip and continue subcommands. BRK and Single Step.
PRICES: CASSETTE $15.95 )at $200 or $3800)EPROM (2716-5V) (at $F000) $49.95 Commented Source $9.95 Custom assembly: add $2.00. User Manual separately: $5.95 (applicable to purchase). All 1st class PPD continental U.S. Other add $2.50

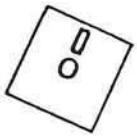
OTHER PRODUCTS: AIM — SYM — KIM
***"Optimized" software for Optimal Technology's EP-2A prommer. Includes erase verify and full prompting. Let your computer do the work. Includes listing, cassette, instructions. Specify system. $9.95
***Complete hard-and software interface for Texas Instruments 12/20 column thermal printer. Requires one eight bit port. Perfect for dedicated control applications. Complete schematics, listings instruction. Specify system. $9.95

J Holtzman
6820 Delmar, 203
St. Louis, MO 63130
314-863-5209



**Progressive Computer Software**

405 Corbin Road
York, PA 17403



*PRESENTS the*

**MACHINE LANGUAGE EXTERMINATOR**

Get rid of those bugs the way the experts do. Our TEMA program allows you to easily locate errors in any machine language program.

ASCII or hex strings can be located, registers and memory locations displayed, plusmany other features that are invaluable for debugging.

Available now on cassette or disk. 48K Apple required.

Cassette—$19.95 Disk—$24.95

APPLE ONLY

| Also Available | Cassette | Disk |
|---|---|---|
| One-Arm Bandit (32K) | 9.95 | 14.95 |
| Card Shark (INT) | 7.95 | 9.95 |
| High Roller (Applesoft) | 7.95 | 9.95 |
| Hi-Res Sub Game (32K) | 14.95 | 19.95 |
| Adult Games Pack | 7.95 | 9.95 |
| Trend LIne Analysis (48K, Applesoft & ROM) | 9.95 | 14.95 |

**CUSTOM PROGRAMMING**

We specialize in custom software. Modifications to existing programs or completely specialized to suit your needs. Write for details.

## Announcing The Micro LIMERICK CONTEST

You could win a free 1-year subscription of MICRO if your limerick is judged to be the best by our staff.

The American Heritage Dictionary defines a limerick as: "A light humerous or nonsensical verse of five lines with the rhyme scheme *aabba*". For those of you who may not have been English majors, here is an example:

*There was a young princess from Niger,*
*Who smiled as she rode on a tiger.*
*They returned from the ride*
*With the princess inside*
*And the smile on the face of the tiger.*

To win our limerick contest, all you have to do is come up with an original 5-line verse about MICRO.

Shouldn't be too tough, right? Winners will be judged on creativity, originality, and above all, humor. Remember to follow the rules of a limerick, and no cheating by using the same word to end the first, second and/or fifth lines! And please, keep them clean.

The winning limerick will be published in the September issue, along with some honorable mention runners-up.

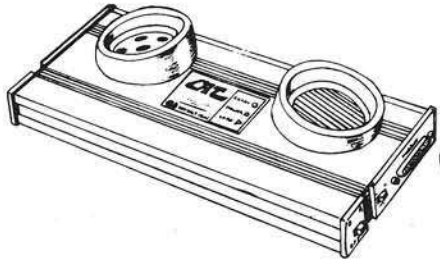
Please send all entries to:

MICRO LIMERICK CONTEST
P.O. Box 6502
Chelmsford, MA 01824

# The Age of Affordable Computing TERMINALS Has Arrived.

HAVE YOU ?

OSI C4P
32 64 CHARACTER LINES $750.00



C A T by NOVATION 300baud $179.00

*SERIAL PORT MODIFICATION of C4P Color by us* $75.00

*A T V Microverter* $35.00

*O S I C 4 P Color Cassette*
*V T 52 Terminal Simulator Prog.* $19.95

*Buy separaœly for* $1,058.95
*(you supply TV and cassette recorder)*

*or your complete checked out as above*

# VT52 $995

See this **OHIO SCIENTIFIC** dealer today!

**COMPUTERSHOP**

Boston
590 Comm. Ave.
(across from B.U.)
247-0700

Cambridge
288 Norfolk St.
(near M.I.T.)
661-2670

# Index, Volume Number 3

## (Issues 13-24)

## (June 1979-May 1980)

### GENERAL


6502 Information Resources Updated — 13:29
William R. Dial

Computer Determined Parameters for Free-Radial Polymerization — 13:38
L.S. Reich

The MICRO Software Catalog:IX — 13:53
Mike Rowe

The MICRO Software Catalog:X — 14:54
Mike Rowe

To Tape or Not to Tape: What is the Question? — 14:57
Noel G. Biles

6502 Bibliography XI — 14:61
William Dial

Replace that PIA with a VIA — 15:17
E.D. Morris Jr.

Tokens — 15:20
E.D. Morris Jr.

The MICRO Software Catalog: XI — 15:38
Mike Rowe

Interfacing the Analog Devices 7570J A/D Converter — 15:40
Marvin L. DeJong

6502 Bibliography XII — 15:53
William Dial

The MICRO Software Catalog: XII — 16:51
Mike Rowe

8080 Simulation with a 6502 — 16:53
Dann McCreary

Writing for MICRO — 16:59
Shawn Spilman

Nicer Writer — 17:5
Rick Connolly

6522 Timimg and Counting Techniques — 17:27
Marvin L. DeJong

The MICRO Software Catalog: XIII — 17:49
Mike Rowe

6502 Bibliography: XIII — 17:61
William Dial

Subroutine Parameter Passing — 18:14
Mark Swanson

Performing Math Functions in Machine Language — 18:30
Alfred J. Bruey

The MICRO Software Catalog: XIV — 18:55
Mike Rowe

6502 Bibliography: XIV — 18:61
William Dial

Sum Test: A Memory Test Routine for the 6502 — 19:73
S. Felton Mitchell Jr.

The MICRO Software Catalog: XV — 19:75
Mike Rowe

6502 Bibliography: XV — 19:77
William Dial

Why a PET, APPLE, 6502 Basic Compiler? — 20:9
Bruce M. Beach

Lifetime of a Non-renewable Resource — 20:21
Marvine L. DeJong

The MICRO Software Catalog: XVI — 20:51
Mike Rowe

6502 Bibliography: XVI — 20:61
William Dial

The Binary Sort — 21:15
Robert Phillips

MICRO Club Forum — 21:29
staff

The MICRO Software Catalog: XVII — 21:55
Mike Rowe

6502 Bibliography: XVII — 21:59
William Dial

Beginning Boolean: A Brief Introduction to Boolean Algebra for Computerists — 22:29
Marvin L. DeJong

Program Checksum Calculator — 22:39
Nicholas Vrtis

The MICRO Software Catalog: XVIII — 22:63
Mike Rowe

6502 Bibliography: XVIII — 22:67
William Dial

The SY6516 Pseudo-16 Bit Processor — 23:36
Randall Hyde

The MICRO Software Catalog: XIX — 23:71
Mike Rowe

6502 Bibliography: XIX — 23:77
William Dial

A Simple Temperature Measurement Program and Interface — 24:19
Marvin L. DeJong

MICRO Club Circuit — 24:66
staff

The MICRO Software Catalog: XX — 24:69
Mike Rowe

6502 Bibliography: XX — 24:73
William Dial


### AIM, SYM, AND KIM


A Symphony in Stereo — 13:17
Phillip M. Rinard

Reading KIM Tapes on the AIM and SYM — 13:34
Robert M. Tripp

AIM 6522 Based Frequency Counter — 13:41
Marvin L. DeJong

KIM — The Tunesmith — 13:43
Anthony R. Scarpelli

SYM-1 Speak to Me — 13:57
Jack Gieryic

AIM Your Spouse Towards Success at the Supermarket 14:19
Melville Evans & Vernon Larrowe
SYM & AIM Memory Expansion 14:30
Paul Smola
The First Book of KIM — On a SYM 14:35
Nicholas Vritis
Extending the SYM-1 Monitor 15:9
Nicholas Vrtis
EPROM for the KIM 15:25
William C. Clements Jr.
SYMple Memory Expansion 15:42
John M. Blalock
An AIM-65 Notepad 16:11
Marvin L. DeJong
SYM-1 Event Timer 16:26
Stephen J. Faris
AIM-65 in the Ham Shack 16:29
Marvin L. DeJong
Tiny Pilot:An Educational Language for the 6502 16:41
Nicholas Vrtis
Card Shuffling Program for the KIM-1 17:41
Hark Chan
SYM-1 6532 Programmable Timer 17:55
Robert A. Peck
Dual Tape Drive for SYM-1 Basic 18:5
George Wells
TSAR: A Time Sharing Administrative Routine (KIM) 18:35
Philip K. Hooper
Interfacing the C1-812 to the KIM 18:43
Jim Dennis
SYM-1 Baudot TTY Interface 18:49
Richard A. Leary
Sharpen Your AIM 19:37
Robert E. Babcock
Time of Day Clock and Calendar for the SYM-1 19:67
Casmir J. Suchyta III and Paul W. Zitewitz
SYM-1 Tape Verification 20:35
Jack Gieryic
Symbol Table Sorter/Printer for the AIM Assembler 20:43
Mel Evans
SYM Staged Loading Technique for Segmented Program 20:59
Robert A. Peck
Expanding the SYM-1 21:5
Robert A. Peck
A Complete Morse Code Send/Receive Package 21:19
Marvin L. DeJong
KIM-1 Tape Recorder Controller 21:35
Michael Urban
Clocking KIM 22:45
Ronald A. Guest
SYMple Basic Data Files 23:21
John Blalock
A Perpetual Calendar Printer for the AIM 23:27
Mel Evans
KIM Scorekeeper 23:59
Joel Swank
SYM-1 Basic "GET" Command 24:7
George H. Wells Jr.
A Formatted Dump Routine for the AIM 24:29
W.E. Wilson
SYM-1 Sends Morse Code 24:49
Ralph R. Orton
Expand Kim-1 Versatility in Systems Application 24:63
Ralph Tenny

**PET**

Case of the Missing Tape Counter 13:11
William F. Pytlik
Reading PET Cassettes without a PET 13:61
David P. Kemp
Boolean Equations Reduced on the PET 14:23
Alan K. Christensen
PET Cassette I/O 15:19
Ronald C. Smith
Plotting a Revolution 16:5
John Sherburne
Move It: Relocating PET Source Programs & Object Code 16:17
Prof. Harvey B. Herman
Speech Processor for the PET 16:35
Charles R. Husbands
Hooking PET to Ma Bell 17:11
C.H. Scanlon
A Tape Indexing System for the PET 18:11
Alan R. Hawthorne
Relocating PET Basic Programs 19:25
Michael Tulloch
An Additional I/O Interface for the PET 19:40
Kevin Erler
A 60 x 80 Life for the PET 19:45
Werner Kolbe
Multiplexing PET's User Port 21:13
Ed Morris
A Machine Language Screen Print Program for the Old or New PET 22:13
Kenneth Finn
Stop That PET! 22:57
Gary Bullard
PET Keysort 23:43
Rev. James Strasma
Plotting with Special Character Graphics 24:11
Dale DePriest
New and Better PET User Port Printer Routines 24:33
Michael Tulloch
Keysort Update 24:58
Rev. James Strasma

**APPLE**

Harmonic Analysis for the APPLE 13:5
Charles B. Putney
Sorting with the APPLE II 13:21
Gary A. Foote
The Color Gun for the APPLE II 13:31
Neil D. Lipson
A Baudot Teletype Driver for the APPLE II 14:5
Lt. Robert Carlson, USN
Intercepting DOS Errors from Integer Basic 14:17
Andy Hertzfeld
Screen Dump to Printer for the APPLE II 14:27
R.M. Mottola
Ampersort 14:39
Alan G. Hill
APPLE II Serial Output Made Simple 15:5
Donald W. Bixby
A Better Life for Your APPLE 15:22
L. William Bradford
What's Where in the APPLE 15:29
Prof. William F. Luebbert
Define HIRES Characters for the APPLE II 15:44
Robert F. Zant
Common Variables on the APPLE II 15:47
Robert F. Zant
Applesoft Renumbering 16:15
J.D. Childress
Life in the Fast Lane 16:21
Richard R. Auricchio
Disassembling the DOS 3.2 17:7
William Reynolds
Spelunker 17:15
Thomas R. Mimlitch

APPLE II HIRES Picture Compression — 18:17
Bob Bishop
Assembly Language Applesoft Renumber — 18:27
Alan D. Floeter
Alarming APPLE — 18:59
Paul Irwin
Data Statement Generator — 19:5
Virginia Brady
How to do a Shape Table Easily and Correctly — 19:11
John Figueras
Applesoft Program Relocation — 19:49
George S. Guild, Jr.
KIM & SYM Format Cassette Tapes on APPLE II — 19:51
Steven Welch
APPLE II Speed Typing Test with Input Time Clock — 19:69
John Broderick
Tape Execute File-Create and Use — 20:5
Allen J. Lacy
Human Physiological Parameters — 20:15
Dr. L.S. Reich
Sweet-16 Programming Using Macros — 20:25
Richard C. Vile, Jr.
Screen Write/File Routine — 20:30
B.E. Baxter
Search/Change in Applesoft — 20:55
J.D. Childress
A HIRES Graph-Plotting Subroutine — 21:9
Richard Fam
APPLE II Floating Pint Utility Routine — 22:7
Harry L. Pruetz
A Digital Thermometer for the APPLE II — 22:21
Carl J. Kershner
A Home Message Center — 22:53
William McLean
Applesoft II Shorthand — 23:5
Allen J. Lacy
The Apple Stripper — 23:11
Bill Crouch
Bi-directional Scrolling — 23:31
Roger Wagner
Roadrunner—A Math Drill for Second Graders — 24:7
Peter A. Cook
Lower Case and Punctuation in Applesoft — 24:45
James D. Childress
An Edit Mask Routine — 24:53
Lee Reynolds


**OSI**


The Basic Morse Keyboard — 13:13
William L. Taylor
Streamlining the C2-4P — 13:28
James L. Cass
Structured Basic Editor and Pre-Processor — 14:7
Robert Abrahamson
OSI Memory Test in Basic — 14:29
William L. Taylor
How Do You Connect Peripherals to Your Superboard II — 17:43
Bruce Hoyt
Hypocycloids — 17:52
E.D. Morris
A Real-Time Clock for OSI Disk Systems — 17:59
Robert T. Kintz
Some Useful Memory Locations and Subroutines for OSI BASIC in ROM — 18:9
S.R. Murphy
If You Treat It Nicely, It Won't Byte — 19:31
Jack Robert Swindell
Graphics and the Challenger C1P — 19:61
William L. Taylor
The Great Superboard Speed Up — 21:31
J.R. Swindell
Graphics and the Challenger C1P, Part 3 — 21:47
William L. Taylor
Polling OSI's Keyboard — 22:17
Edward H. Carlson
Challenger II Cassette Techniques — 22:25
Richard A. Lary
Graphics and the Challenger C1P, Part 4 — 23:15
William L. Taylor
OSI Basic in ROM—What's Where? — 23:65
Earl D. Morris Jr.
Shorthand Commands for Your OSI Superboard II — 24:25
Henk J. Wevers
Graphics and The Challenger C1P Part 5 — 24:41
William L. Taylor


**Editor's Note:** Use this index as a guide to material which is of interest to you. Remember that most of the articles and features under the 'General' heading apply to 6502 systems in general. Also, many articles and programs which are classified for one system may be readily modified and/or adapted to another system.


## Missing MICRO Information?

MICRO is devoted exclusively to the 6502. In addition, it is aimed at useful, reference type material, not just "fun and games". Each month MICRO publishes application notes, hardware and software tutorials, a continuing bibliography, software catalog, and so forth. Since MICRO contains lots of reference material and many useful program, most readers want to get the entire collection of MICRO. Since MICRO grew very rapidly, it quickly became impractical to reprint back issues for new subscribers. In order to make the older material available, collections of the reprints have been published.

[A limited number of back issues are still available from number 7 to 18 and 20 to current. There are no 19's left. Use the order form in this issue.]

For a free copy of the Index for Volumes 1 and 2, please send a self-addressed, stamped envelope to:
**BEST of MICRO, P.O. Box 6502, Chelmsford, MA 01824**

**The BEST of MICRO Volume 1** contains all of the significant material from the first six issues of MICRO, covering October/November 1977 through August/September 1978. This book form is 176 pages long, plus five removeable reference cards. The material is organized by microcomputer and almost every article is included. Only the ads and a few 'dated' articles have been omitted. **Surface . . . $7.00**
[Now in third printing!] **Air Mail . . . $10.00**

**The BEST of MICRO Volume 2** covers the second six issues, from October/November 1978 through May 1979. Organized by microcomputer, this volume is 224 pages long. **Surface . . . $9.00**
**Air Mail . . . $13.00**

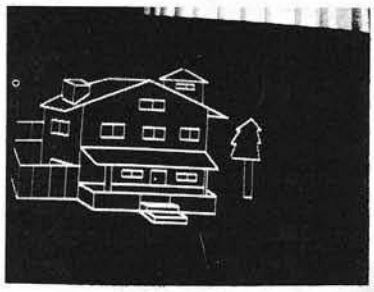
**The BEST of MICRO Volume 3,** covering the twelve issues from June 1979 through May 1980, will be over 400 pages long. It is scheduled for late summer 1980. The price is still to be determined.

# NEW APPLE II / APPLE II PLUS SOFTWARE FROM ON– LINE SYSTEMS

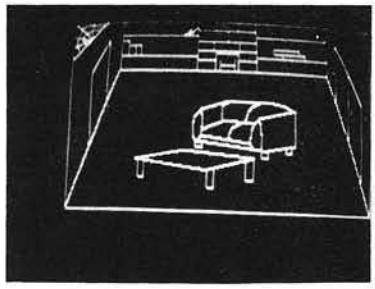
TIRED OF BUYING GAMES THAT BECOME BORING AFTER A FEW HOURS OF PLAY? ON–LINE SYTEMS IS DEDICATED TO DELIVERING SERIOUS SOFTWARE FOR THE DISCRIMINATING GAMESMAN. THESE PRODUCTS HAVE BEEN SIX MONTHS IN DEVELOPMENT AND PROVIDE THE QUALITY AND SPEED POSSIBLE ONLY THROUGH MACHINE LANGUAGE!

## *ALL NEW*

HI-RES ADVENTURE (''MYSTERY HOUSE'')



What is an adventure game? According to the dictionary, an adventure is a hazardous or daring enterprise; an exciting experience; to risk, hazard, to venture on. One who goes on an adventure is a venturer. A seeker of fortune in daring enterprises; a speculator. In essence, an adventure game is a fantasy world where you are transported, via your own computer. You are the key character of the fantasy as you travel through a land the likes of which you will find in books that take you, through your imagination, to the world it is creating.



Through the use of over a hundred Hi-Res pictures you play and see your adventure. You communicate with HI-RES ADVENTURE in plain english (it understands over 300 words!) All rooms of this spooky old house appear in full Hi-Res Graphics complete with objects you can get, carry, throw, drop, or ?.

In this particular HI-RES ADVENTURE game, you are transported to the front yard of a large, old victorian house. When you enter the house you are pulled into the mystery, murder, and intrigue and can not leave until you solve the puzzles. Your friends are being murdered one by one. You must find out why, and who the killer is. Be careful, because the killer may find you! As you explore the house there are puzzles to be solved and hazards to overcome. The secret passage-way may lead you to the answer.

### *ALSO NEW FROM ON–LINE SYSTEMS*

SKEET/TRAP have become Olympic shooting sports and and obsession among Scatter-gunners all over the world. These games are the All-American although they have become international.

SKEETSHOOT allows one to five shotgunners to test their marksmanship as they fire from the eight prescribed positions on an official NSSA skeet firing range. Each position provides a new perspective of the field with the pigeons travelling at different angles. At each position a pigeon is launched from one side of the field and then the other. At certain positions, pigeons are launched from both sides of the field simultaneously. This is a true game of skill, simulating skeet shooting down to the last detail.

TRAPSHOOT allows one to five shotgunners to test their markmanship. The trap firing range has five positions where the one to five players shoot from. Each player is at a different location on the field. The challenge is to shoot pigeons out of the sky which launch at random trajectories. The challenge is to hit the pigeons while they are still in gun range.

SKEETSHOOT and TRAPSHOOT both allow you to control the size and speed of the pigeons and the width of your shotgun spray. Realistic sound-effects and HI-RES animation combine to make this simulation unparrelled for the APPLE.

**DEALER INQUIRIES INVITED!**

**ALL SOFTWARE SHIPPED SAME DAY.**
**PHONE ORDER: (805) 522-8772**

**ON–LINE SYSTEMS, 772 NO HOLBROOK, SIMI, CA 93065**

**SEND TO:** _____

_____

| | | QUANTITY | TOTAL |
|---|---|---|---|
| Hi-Res Adventure/Disk | $24.95 | | |
| Skeetshoot/Cassette | $14.95 | | |
| Trapshoot/Cassette | $ 9.95 | | |
| Skeetshoot/Disk | $19.95 | | |
| Trapshoot/Disk | $14.95 | | |
| Hi-Res Adventures & Skeetshoot/Disk | $37.50 | | |
| | | Subtotal | |
| Payment: Check_____ | | 6% tax (Calif.) | |
| Master Chg/Visa #_____ | | Shipping | $1.00 |
| Expires:_____ | | Total | |

NOW PRESENTING...

# Software for Apple® II

## for your Entertainment · Business · Education

# Star Attractions:

**+WRITE-ON!** Professional Word Processing lets you edit, move, delete, find, change and repeat any body of text, merge and save on disk. Does right-justified margins, centering, page numbering. You can enter name & address onto form letters when printing. Edit and merge any text disk file—even files not created by WRITE-ON—and spool text to disk for letter printing or editing. Chain up to 100 files in a single printer run. Needs Applesoft and 32K.
**On Disk with operating manual** ..... **$99.50**

**FILEMASTER I** Here's a powerful data file manager giving you two programs—FORMAT & RETRIEVAL. It handles everything from phone lists to legal abstracts. You can design your own data structure with up to 500 characters per record and up to 15 searchable fields in any combination. Needs 32K. ***On Disk*** ..... **$49.95**

**FILEMASTER II** Has all the same features of FILEMASTER I plus allows for totaling, advanced math routines, more powerful print formatting, larger data fields, and disk-to-disk transfers ..... **$99.50**

**+Space** (Edu-Ware) Six programs form a unique epic game series. Multi-faceted simulation of life in interstellar society. You and opponents must make life & death decisions. Keeps track of your progress from one game to next. Needs 48K and Applesoft ROM. **Disk** ..... **$29.95**

**Pot O'Gold I or our All New Pot O' Gold II.** A collection of 49 programs for 16K Apple. Everything from Logic to action games. Only a buck a game. Specify I or II.
Price each: **Tape** ..... **$49** **Disk** ..... **$54**

**+Adventure** This original, full-function game is the same as the one developed for large mainframes. Fight off pirates and vicious dwarfs. 700 travel options, 140 locations, 64 objects. Needs Applesoft ROM & 48K. **Disk** ..... **$29.95**

**32K Disk Inventory:** Use stock numbers, description, vendor, record of purchase and sales date, amount on hand, cost & sell price, total value. Holds up to 300 items.
**Disk** ..... **$40**
+With Bill of Materials: **Disk** ..... **$50**

**32K Data Base** Cross file for phone lists, bibliographies, recipes. Run up to 9 lines of 40 columns each. Search by item anywhere.
**Disk** ..... **$20**

**24K Hi-Res Life Simulation** Conway's equations on 296x180 screen. A mathematical simulation to demo population growth with birth, death and survival as factors. **Tape** ..... **$10**

**16K Rainbow's Casino** 9 gambling games: Roulette, Blackjack, Craps, Horserace, Yahtzee, Keno, Slot Machine, Poker, and Acey-Ducey. Needs 16K. **Tape** ..... **$29.95** **Disk** ..... **$34.95**

**16K Space War:** You in your space capsule battle against the computer's saucer . . . in hi-res graphics. **Tape** ..... **$12**

**16K Memory Verify** Diagnostic routine to check range of memory. Indicates faulty addresses, data in memory cell, and faulty data. **Tape** .... **$5**

**16K Appleodion** Music synthesis composes original Irish jigs. Enter your own music and save on tape or disk. Includes 3 Bach fugues.
**Tape** ..... **$10**

**+48K Edu-Pack** (Edu-Ware) This package combines COMPU-READ—five speed reading programs; three PERCEPTION games where random shapes and sizes must be matched; and STATISTICS for computing Mean, Variance, Standard Deviation, and much more! Needs Applesoft ROM. **Disk** ..... **$39.95**

**+32K Sargon II** (Hayden) Here's the program that came in third against the big machines (mainframes and maxis) at the 9th North American Computer Chess Championship and placed first in the European Microcomputer Chess Championships! Has seven levels of play with Levels 0 - 3 playing in tournament time. Need a challenge? This is it!!
**Tape** ..... **$29.95** **Disk** ..... **$34.95**

**16K Hi-Res Baseball** (Programma International) This animated simulation of a major league baseball game is for *two players*. The scoreboard is in the lower left of screen with the "throw pointer" for directing a throw in the lower right corner. Written entirely in machine language, the action is quick and smooth, making it the finest simulation of its kind.
**Tape** ..... **$15.95**

**OTHER SPECIAL ITEMS FOR YOUR APPLE II**

**VERSAWRITER** This digitizer drawing board, complete with a powerful software package on disk, lets you create any picture in color with high-resolution graphics. It's ideal for mass graphics. You can trace, edit, save and recall what you draw. It's a simple-to-use system for students, artists, engineers and graphic programmers. Has an 8½"x11" working area. New applications added include: •Text Writer adds text to your pictures. You control size, color and direction of text; •Electronic Drawing lets you create schematics and includes commonly used symbols for transistors, OPAMPS, and FETS; •Distance/Area lets you compute distances on maps or area of any frame. Applesoft ROM and 32K required. ..... **$249.95**
*Add $5 (U.S.) or $10 (Foreign) for shipping.*

**Apple Monitor Peeled** Everything you wanted to know about the Apple Monitor but couldn't figure out. All the PEEKS, POKES, and CALLS explained in an easy-to-understand form, written in plain English.
**Only** ..... **$9.95**

**Don't see what you've been looking for, here? Then write for our FREE SOFTWARE CATALOG. We're saving one just for you!**

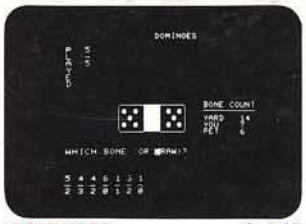
To order, add $2 (U.S.): $5 (foreign) for shipping. California residents add 6% sales tax. Sorry, we cannot ship to P.O. Boxes. VISA/MASTERCHARGE and BANKAMERICARD Welcomed!

RAINBOW COMPUTING INC.

Golden Plaza Shopping Center, Dept. 1A
9719 Reseda Blvd., Northridge, CA 91324
Telephone: (213) 349-5560

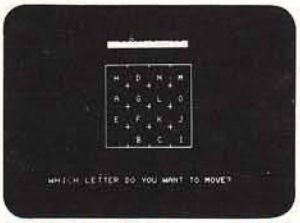
+ = Apple Plus compatible

# Software for the PET



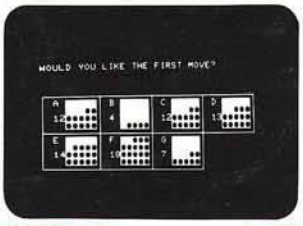

DOMINOES $ 6.95

OTHELLO $ 9.95

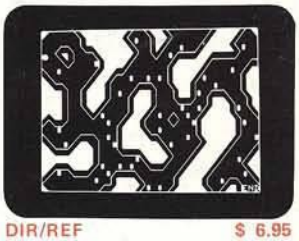



LETTER SQUARES $ 6.95

RPN MATHPACK $19.95



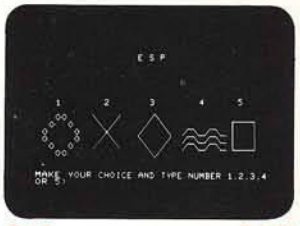

SUPER NIM $ 6.95



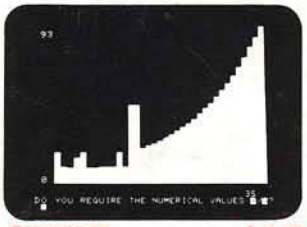
DIR/REF $ 6.95

SPACE WARS $ 9.95

GRAND PRIX $ 6.95

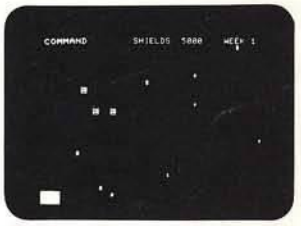



E.S.P. $ 9.95

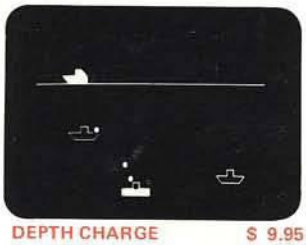



FORECAST $ 9.95

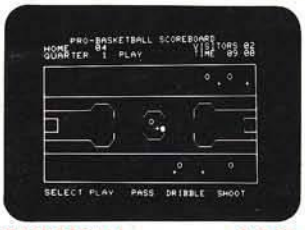
FOOTBALL $ 6.95



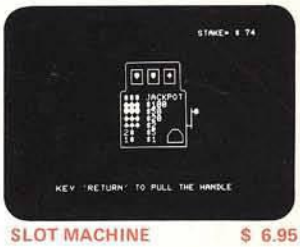

BLOCKADE $ 9.95



DEPTH CHARGE $ 9.95




BASKETBALL $ 9.95



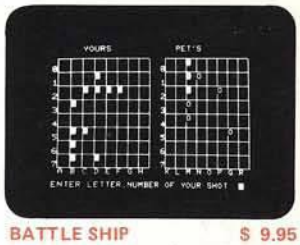

SLOT MACHINE $ 6.95

HOME ACCOUNTING $ 9.95

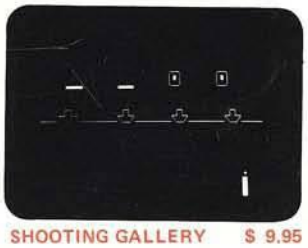
All orders include 3% postage and handling with a minimum of $1.00. California residents include 6% Sales Tax.

**VISA** **MASTERCHARGE**

PET IS A TRADEMARK OF
COMMODORE BUSINESS MACHINES, INC.




BATTLE SHIP $ 9.95



SHOOTING GALLERY $ 9.95


**PROGRAMMA INTERNATIONAL, Inc.**
3400 Wilshire Blvd.
Los Angeles, CA 90010
(213) 384-0579
384-1116
384-1117


Dealer Inquiries Invited

ZAP $ 6.95

PROGRAMMA

**Software Program Products**